The Dissertation Committee for Chetan Kumar Jhurani
certifies that this is the approved version of the following dissertation:

Multiscale Modeling Using Goal-oriented Adaptivity and Numerical Homogenization

Committee:

Leszek F. Demkowicz, Supervisor

Todd Arbogast

J. Tinsley Oden

Robert A. van de Geijn

C. Grant Willson

Multiscale Modeling Using Goal-oriented Adaptivity and Numerical Homogenization

by

Chetan Kumar Jhurani, B.Tech.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2009

To my family.

Acknowledgments

This dissertation would not be possible and the desire to work toward it would be meaningless without the presence and support of my family, mentors, teachers, friends, relatives, and colleagues over the years.

I would like to thank Professor Leszek Demkowicz for his professional guidance, academic advice, and vision toward the goal. But most importantly I thank him for his patience, care, and constant encouragement.

It has been a privilege to work with Professor J. Tinsley Oden in this project. His desire and efforts to get to the truth are unmatched. Professor C. Grant Willson provided the physical problem and has been very helpful and encouraging throughout the project. The help of the dissertation committee formed by Professor J. Tinsley Oden, Professor C. Grant Willson, Professor Todd Arbogast, and Professor Robert A. van de Geijn has been crucial and the discussions with them have been fruitful.

It has been a very good experience working and interacting with Paul T. Bauman, David Fuentes, Jason Kurtz, Christian Michler, Serge Prudhomme, and Elizabeth Collister. I would also like to thank all the participants of the group meetings for Multiscale Modeling and *hp*-adaptivity for their critical feedback.

It takes a department to complete a Ph.D. My thanks go to Cay Garcia, Sue Hennigar, Donna Larson, Charlott Low, Stephanie Rodriguez, Lorraine Sanchez, Kathleen Sparks, and the ICES Sysnet team for their administrative help.

If it were not for the contributors to all the free software I have used, this work would have been more work and less fun.

This work has been supported by a Department of Energy grant, a Teaching Assistantship by the Department of Aerospace Engineering and Engineering Mechanics, and a Fellowship from University of Texas at Austin.

In the past, my teachers, mentors, and colleagues from Choithram School (Indore), Indian Institute of Technology (Bombay), Geometric Software, and McAfee Inc. have been very helpful. Specifically, I would like to express my gratitude to Mrs. Neeta Arora, Professor S. Ghosh Moulic, Professor Ravi S. Nanjundiah, Yogesh Sajanikar, and Dr. Shailesh Deshpande.

I wish to thank Amanvir, Aniket, Aruna, Ashish, Bala, Bhoopesh, Carla, Dolly, Easwar, Hirendra, Kapil, Neely, Pankaj, Preethi, Priya, Rachna, Rajani, Raju, Rashmi, Richa, Richie, Sandeep, Sanjeev, Saurabh, Sena, Shailendra, Shikha, Shilpa, Shreyas, Shubra, Siddhartha, Smita, Ujjwal, Ulka, Vidhya, and Vishy for their support and giving me some of the best times.

Finally, I would like to remember my family. My parents, my brother Roopesh, all members of Jhurani family, Tahilramani family, and Gulati family have supported me in every possible way. My wife Shilpa has been a constant

source of love, encouragement, conversations, insights, and last but not the least, delicious food. I am eternally grateful to all of you.

Chetan Kumar Jhurani

The University of Texas at Austin

August 2009

Multiscale Modeling Using Goal-oriented Adaptivity and Numerical Homogenization

Publication No. _____

Chetan Kumar Jhurani, Ph.D.
The University of Texas at Austin, 2009

Supervisor: Leszek F. Demkowicz

Modeling of engineering objects with complex heterogeneous material structure at nanoscale level has emerged as an important research problem. In this research, we are interested in multiscale modeling and analysis of mechanical properties of the polymer structures created in the Step and Flash Imprint Lithography (SFIL) process. SFIL is a novel imprint lithography process designed to transfer circuit patterns for fabricating microchips in low-pressure and room-temperature environments. Since the smallest features in SFIL are only a few molecules across, approximating them as a continuum is not completely accurate. Previous research in this subject has dealt with coupling discrete models with continuum hyperelasticity models. The modeling of the post-polymerization step in SFIL involves computing solutions of large nonlinear energy minimization problems with fast spatial variation in material

properties. An equilibrium configuration is found by minimizing the energy of this heterogeneous polymeric lattice.

Numerical solution of such a molecular statics base model, which is assumed to describe the microstructure completely, is computationally very expensive. This is due to the problem size – on the order of millions of degrees of freedom (DOFs). Rapid variation in material properties, ill-conditioning, nonlinearity, and non-convexity make this problem even more challenging to solve.

We devise a method for efficient approximation of the solution. Combining numerical homogenization, adaptive finite element meshes, and goal-oriented error estimation, we develop a black-box method for efficient solution of problems with multiple spatial scales. The purpose of this homogenization method is to reduce the number of DOFs, find locally optimal effective material properties, and do goal-oriented mesh refinement. In addition, it smoothes the energy landscape.

Traditionally, a finite element mesh is designed after obtaining material properties in different regions. The mesh has to resolve material discontinuities and rapid variations. In our approach, however, we generate a sequence of coarse meshes (possibly 1-irregular), and homogenize material properties on each coarse mesh element using a locally posed constrained convex quadratic optimization problem. This upscaling is done using Moore-Penrose pseudoinverse of the linearized fine-scale element stiffness matrices, and a material independent interpolation operator. This requires solution of a continuous-time

Lyapunov equation on each element. Using the adjoint solution, we compute local error estimates in the quantity of interest. The error estimates also drive the automatic mesh adaptivity algorithm. The results show that this method uses orders of magnitude fewer degrees of freedom to give fast and approximate solutions of the original fine-scale problem.

Critical to the computational speed of local homogenization is computing Moore-Penrose pseudoinverse of rank-deficient matrices without using Singular Value Decomposition. To this end, we use four algorithms, each having different desirable features. The algorithms are based on Tikhonov regularization, sparse QR factorization, *a priori* knowledge of the null-space of the matrix, and iterative methods based on proper splittings of matrices. These algorithms can exploit sparsity and thus are fast.

Although the homogenization method is designed with a specific molecular statics problem in mind, it is a general method applicable for problems with a given fine mesh that sufficiently resolves the fine-scale material properties. We verify the method using a conductivity problem in 2-D, with chessboard like thermal conductivity pattern, which has a known homogenized conductivity. We analyze other aspects of the homogenization method, for example the choice of norm in which we measure local error, optimum coarse mesh element size for homogenizing SFIL lattices, and the effect of the method chosen for computing the pseudoinverse.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

We describe a physical problem with multiple spatial scales, and a computationally efficient solution technique for its mathematical model. A brief literature review of multiscale methods, numerical homogenization, molecular mechanics, adaptivity and error estimation for elliptic problems, and algorithms for computing Moore-Penrose pseudoinverses is presented. Finally we describe a goal-oriented and adaptive numerical homogenization method for nonlinear optimization problems in heterogeneous physical media.

1.1 Motivation

All physical media are heterogeneous. Any homogeneous continuum description stops being valid when the spatial scale reduces to atoms and molecules. Even before reaching the smallest scale, engineering materials exhibit a hierarchy of scales with different mechanical structure, properties, and physical models. Multiscale or multilevel modeling and analysis methods try to capture and describe these different scales in relation to each other.

Traditionally, however, we work at a single spatial or temporal scale. Other scales are either not important or are explicitly ignored. Effective mate-

rial properties of a coarse-scale of interest are derived using physical or numerical experiments on a Representative Volume Element that is much larger than the small scales but still smaller than the large scales [82]. This approximation is justified for many problems when we are not interested in processes at smaller scales. Even if one could simulate all the scales together, the problem of acquiring input for the model is practically infeasible. In case of numerical experiments, such an approximation is justified since it is simpler and limited computational resources prohibit simulation of all the scales.

In last few decades, the technology to create small structures has improved to reach nanoscale objects. Research in semiconductor technology has been an important driving force in this progress [60]. Step and Flash Imprint Lithography (SFIL) is an imprint lithography process designed to transfer circuit patterns to fabricate microchips in low-pressure and room-temperature environments [10, 29, 9]. It has enabled imprinting of features smaller than 20 nanometers (nm). It has the inherent resolution necessary to define sub-10nm geometries [75]. The main components of the SFIL process are the template, etch barrier, transfer layer, and substrate. The process copies features from the template onto the substrate by photopolymerizing the ultraviolet-sensitive etch barrier solution. Photopolymerization is accompanied by densification which affects the shape of imprinted features [30].

In this research, we are interested in the post-polymerization step of the SFIL process. The object of interest is a glassy polymeric structure created on an organic polymer layer which in turn is on a silicon substrate. We want

to model the changes in the mechanical structure of the polymeric pattern, also referred to as densification. Typical dimensions of the patterns in the structure are much larger than individual molecules, but the discreteness plays an important role in modeling of such objects. In the context of SFIL, an approach for coupling of discrete polymer elasticity models with continuum hyperelasticity models has been presented in [14, 15].

Rapid increase in computational power has allowed large simulations at atomistic and molecular level. Still, the capability of current computers poses a limit on the size of such problems. Moreover, we are not interested equally in every atom or molecule. An alternative approach is to find effective properties of a group of molecules in some locations and work with fine-scale properties where necessary. Such techniques of homogenization are frequently used to model a heterogeneous continuum medium as a relatively homogeneous medium on a coarser scale [82].

To efficiently simulate the large number of molecules forming the polymer chains, we have to adapt the existing techniques of homogenization and create new ones suitable to this problem. Our objective is to approximate a nonlinear base model of the polymeric structure (molecular statics) by local numerical homogenization and use goal-oriented adaptivity to change the models spatially.

1.2 Multiscale methods and numerical homogenization

The desire to capture fundamental or more accurate small-scale models into large-scale models has given rise to the field of multiscale methods [44]. Their applicability is shown in a wide range of fields where understanding of heterogeneity is critical, for example in material sciences.

Resolution of fine scales is important for accuracy of the computed solution. This is a demanding task because the coarse scales could be orders of magnitude larger than the fine scales. This makes a typical multiscale problem intractable unless the subgrid properties are taken into account using auxiliary small problems. Hence, multiscale analysis is an active research area and many numerical homogenization methods have been proposed recently. Such methods find the approximate solution on a coarse mesh but use the fine grid to construct the relevant information. They do not rely on the assumption of periodicity that is typically used in classical homogenization.

Hughes et al. use concepts of variational multiscale and residual-free bubbles to resolve the fine-scales [51, 26]. Engquist et al. use wavelet basis to compute effective homogenized operators and use truncation for a sparse approximation [43, 45]. Hou et al. compute operator-dependent basis functions by solving local auxiliary problems [50]. For two-phase flow in porous media, a numerical upscaling technique based on the assumption that net flux between coarse elements occurs only on the coarse-scale has been introduced by Arbogast [5]. In [58], Knapek introduced operator-dependent interpolation in the context of multigrid methods.

The methods stated above are used on the continuum level. In molecular simulations, the fine-scale is made up of discrete entities but the coarse-scale can be discrete or a continuum. If the coarse-scale is a continuum, additional sophistication is needed to make the two models mathematically compatible. Hence, in any coupling procedure of two models like these, extra care has to be taken at the interface(s). This can be done by choosing an overlap for transfer of information and enforcing continuity weakly. Many methods have been developed that take care of such problems, for example the Arlequin method to couple particles and continuum [14, 15, 16], and the bridging domain method of Xiao and Belytschko[81].

1.3 Molecular mechanics

Although the power of molecular models cannot be overestimated, their simulation in itself is a challenging problem. The physical parameters required to even pose the problem are hard to get, whether by *ab initio* quantum mechanical simulation, or an empirical method. After all this, prediction of the minimum energy molecular structure is a highly non-convex optimization problem with a large number of local minima [23]. An assumption frequently made is that the global minimum is the important and desired one [64]. Location of the global minimum is obviously special, but whether it is reached in physical systems is difficult to ascertain. For non-convex problems, the location of the global minimum is typically a discontinuous function of the input data. Hence, solving for the location is ill-posed in the sense of Hadamard. This is

reflected in the fact that there does not exist any computationally competitive and general-purpose algorithm for finding global minima of non-convex functions. Nonetheless, a wide range of techniques have been tried for global minimization of molecular structures [76, 23].

1.4 Adaptivity and error estimation for elliptic problems

Critical to the accuracy, reliability and mesh adaptivity in finite element methods is existence of good *a posteriori* error estimates. Such estimates not only provide confidence in the solution on the current mesh but also indicate elements to be refined further for an automatic refinement strategy. Use of effective automatic refinement algorithms is essential to obtain an accurate solution of problems in complex domains.

There have been many classes of *a posteriori* error estimators to estimate the global error due to finite element approximation. Each estimator utilizes the solution on the current mesh in a different way. Babuška and Rheinboldt use the residual in the differential equation on each element and jumps in normal derivatives across element boundaries to compute local error indicators [8]. Other approaches involve solving auxiliary problems on each element. Bank and Weiser solve a Neumann problem on each element with the load coming from the residual and jumps on the element interface [12]. The finite element space for the local problem uses polynomials of degree higher than those of the original space. A similar approach, called the *element resid-*

ual method, was used by Oden et al. [68] for *a posteriori* error estimation. In another paper [7], Babuška and Rheinboldt estimate the error by solving local Dirichlet problem with boundary data coming from the current approximate solution. The local finite element space is enriched by increasing the polynomial degree. The difference of the two solutions provides a local estimate.

Many researchers have contributed to the vast field of *a posteriori* error estimation and mesh refinement. For a comprehensive introduction and analysis of various methods, we refer to the monographs by Babuška and Strouboulis [6] and by Oden and Ainsworth [1].

For many applications, interest is restricted to part of the full domain or a goal represented by a functional of the solution. Usually, in the context of linear problems, the goal is a bounded linear functional on the Hilbert space containing the solution. Recently, many algorithms have been developed for optimizing the mesh for reducing the error in a given quantity of interest rather than in some energy norm. Such algorithms provide the basis for the so-called goal-oriented adaptivity. The main tool behind such algorithms is characterization of the error in the goal in terms of the solution of the adjoint problem (which is driven by the goal). Amongst others, this approach was taken by Becker and Rannacher [17] and Oden and Prudhomme [74].

These efforts for *a posteriori* error estimation work for h-adaptivity. In the field of higher order finite elements and automatic hp-adaptivity, a uniformly h and p refined grid has been used to decide between local h or p refinements [38, 59]. The fine grid solution is a much better approximation

to the exact solution and eliminates the need of explicit *a posteriori* error estimation. Goal-oriented adaptivity has also been extended to work with this two-grid paradigm. It delivers accurate local error indicators and converges exponentially [39, 71].

1.5 Computing Moore-Penrose pseudoinverse

Moore-Penrose pseudoinverse was discovered by E. H. Moore in 1906 in the context of projections associated with singular and rectangular matrices [63]. It was rediscovered by Penrose in 1955 as the unique matrix satisfying four algebraic matrix equations [73]. Quite a few articles and books have appeared on the subject since then. There is a large bibliography in [67] and [19].

For an arbitrary dense matrix, Singular Value Decomposition (SVD) is the most reliable algorithm to compute the pseudoinverse (or its action on a vector). It is important to determine (or know) the numerical rank of a matrix. This is because of the unusual perturbation properties of the pseudoinverse [48]. For this reason, general purpose linear algebra packages like MATLAB® use the SVD to compute the pseudoinverse. But the cost of computing the SVD is high. It is $O(n^3)$ where n is the matrix size (assuming a square matrix). However, SVD is useful for determination of the numerical rank, which is important to reliably compute pseudoinverse of arbitrary matrices.

Despite its excellent numerical properties, use of SVD gives an impression that computation of pseudoinverse is *inherently* an iterative process just

like the computation of singular values or eigenvalues. This impression is not correct. Entries of the pseudoinverse matrix are explicit rational functions of the entries of the original matrix just like the entries of the inverse of an invertible matrix. This is readily seen from the equivalent definition of the pseudoinverse that uses Tikhonov regularization [19] or full-rank factorizations. Looking just at Tikhonov regularization, the entries of the inverse of a non-singular matrix are rational polynomials in the entries of the original matrix. Thus, the entries of the pseudoinverse matrix are limits of rational polynomials expressions. In principle, they can be expressed as explicit functions and are exactly computable unlike eigenvalues or singular values. We do not use this observation for computing pseudoinverses, but mention it because of its conspicuous absence in the literature. Moreover, this gives a hint that if we know the rank of the matrix *a priori*, possibly using physical considerations, then better algorithms can be devised, for example the one in [47].

As is common in the field of computational mathematics, much better algorithms can be developed if one has additional information about the problem input, or if one needs only an approximate solution, or if one has to repeatedly compute something with moderate changes in data. All these cases are possible for computing pseudoinverses in the context of multiscale methods.

Since pseudoinverses are generalizations of inverses of square nonsingular matrices, their computation usually involves solving linear equations or obtaining matrix factorizations. Analogous to the plethora of direct and iter-

ative algorithms for solution of linear systems of equations, there is no single algorithm that is the best choice for computing generalized inverses. There is a variety of algorithms that can be used [19]. Moore-Penrose pseudoinverse can be computed using full-rank factorization. The pseudoinverse of the original matrix can be computed using pseudoinverses of the factors. There are other direct algorithms too, for example Greville's method discussed in [19]. We focus only on a few that are most likely to be useful for our purposes. These are presented as part of the next section and in detail in Chapter 5.

1.6 Scope of this work

In this dissertation, we desire to develop and implementation of a framework for numerical homogenization and goal-oriented adaptivity for nonlinear lattice elasticity problems. The method is developed with the polymer base model of SFIL in mind, but is quite general and can be applied to continuum problems with a given fine mesh that sufficiently resolves the fine-scale material properties.

We describe the SFIL process, its modeling, and parameter estimation in Chapter 2. In Section 2.5, we present the experimental method for estimating the bond parameters of the fine-scale lattice. In Chapter 3, we describe the use of automatic *hp*-adaptivity to generate meshes suitable for approximating solutions of linear lattice elasticity problems with smoothly varying material properties. In Chapter 4, we present a local numerical homogenization method based on Moore-Penrose pseudoinverse of element stiffness matrices.

We present fast algorithms for Moore-Penrose pseudoinverse of sparse singular matrices in Chapter 5. Finally, in Chapter 6 we show numerical results of the overall algorithm, which requires fast computation of pseudoinverses, integration of local numerical homogenization, goal-oriented adaptivity, and iterative steps of a nonlinear optimization method.

Currently there is no general analytic or numerical technique with quantifiable errors for homogenizing nonlinear elastic lattices. This research develops two different methods to achieve this goal.

In the first approximation method, presented in Appendix B, we use a formal limit of the lattice energy and derive a lattice-dependent continuum hyperelasticity model. The derived continuum model can be used on its own or can be coupled with the exact lattice model in areas where higher accuracy is desired.

The second method, which is the focus of this work, is detailed in Chapters 4, 5, and 6. We use the theory of homogenization of continuous periodic media as a guideline to obtain optimal effective properties of a material with a discrete microstructure. The purpose of homogenization is to find local effective properties and reduce number of DOFs for the global problem. In addition, homogenization smooths the energy landscape. We also present theoretical justifications for the choice of this method. Numerical experiments prove the validity of our claims. An automatic goal-oriented adaptive mesh h-refinement algorithm reduces the error due to coarsening.

The numerical homogenization method is related to G-convergence of differential operators [78, 28] in the sense that it appropriately "averages" an inverse to define the homogenized operator. The philosophy is similar to various other multiscale approaches that upscale local information by solving local problems, for example numerical subgrid upscaling [5], multiscale finite element methods [50], and multigrid homogenization [58].

Traditionally, a finite element mesh is designed after obtaining material properties in different regions. The mesh has to match material discontinuities. In our approach we do exactly the opposite. A coarse mesh is selected first and homogenization is done on each individual element of the mesh. In this way, the coarse mesh and the fine-scale structure become naturally compatible. On each element, the homogenization method works with the Moore-Penrose pseudoinverse of the element stiffness matrix to produce pseudoinverse of the local homogenized stiffness matrix as the output. This output depends on the local load as well as a chosen local interpolation from the coarse to the fine mesh. This process requires solving a continuous-time Lyapunov equation on each element. The pseudoinverse of the homogenized stiffness matrix is then pseudo-inverted to get the homogenized stiffness matrix for the assembly phase. Computing homogenized stiffness matrices can be done on each element independently of other elements. Use of the adjoint solution provides local error estimates in the quantity of interest. These estimates indicate which elements should to be h-refined for the next mesh. This process continues until the error estimate in goal is below a required tolerance.

Local homogenization is the core part of the method and is done quite frequently because of mesh refinement as well as nonlinearities. Critical to the efficiency of the local homogenization is fast and approximate computation of the pseudoinverse of a sparse element stiffness matrix. Thus, Singular Value Decomposition is not the best choice. We present algorithms that are significantly faster for sparse matrices. The first method uses the characterization of the pseudoinverse as the limit of a Tikhonov regularized sequence [79, 19]. In the second method, the knowledge of the null-space of a matrix can be used to compute the pseudoinverse by direct or iterative linear solvers [47]. The third method uses a sparse rank-revealing QR factorization of the matrix (or of a suitable column permutation) to compute an "exact" pseudoinverse using the matrix factors Q and R [24, 34]. Lastly, in the context of mesh refinement or nonlinear problems, one might reuse an old pseudoinverse (actually its factorized form) to compute pseudoinverse of a perturbed matrix. This can be done using an iterative procedure based on 'proper splittings' [22, 66]. All these algorithms are discussed in Chapter 5.

The work extends the existing goal-oriented h-refinement strategy [39] to numerical homogenization where coarse-scale and fine-scale operators are different. The adjoint solutions on coarse and fine meshes provide a basis of automatic goal-oriented adaptivity. This gives rise to 1-irregular meshes with hanging nodes. These are handled using the constrained approximation techniques [37, 35].

Chapter 2

Description and Modeling of Step and Flash Imprint Lithography

Advances in fabrication at nanoscale have led to faster computing chips in the last few decades. A number of technologies are available and many are being developed for creating and replicating structures at the molecular level [60, 32] . Optical lithography, electron beam lithography, X-ray lithography, and variations of nanoimprint lithography are a few examples. Within these choices, optical lithography has remained the dominant technology for creating circuit patterns on chips since the inception of the field. The wavelength of ultra-violet (UV) light has characterized the size of the smallest features. This limit has been reached, and faster techniques for smaller structures are being developed in response to the demands of the market.

Step and Flash Imprint Lithography (SFIL) is a viable low-cost alternative to existing lithography techniques. This technique was initially developed by the Willson Research Group at The University of Texas at Austin in the late 1990s [29]. It is designed for fabricating microchips in low-pressure and room-temperature environments [10, 29, 9] . It has enabled imprinting of features smaller than 20 nanometers (nm). Moreover, it has the inherent resolution

necessary to define sub-10 nm geometries [75] . Roughly speaking, a template contains "negative" of the desired pattern. If liquid were to be trapped inside these negative features and polymerized, on removal of the template, we would obtain a polymer with the "positive" pattern. Figure 2.1 shows the basic idea behind the process and the resulting geometry. This chapter describes the process, the existing model of elasticity of polymeric lattices created in SFIL, and parameter estimation for bond potentials. The lattice elasticity model is described in detail in [14, 15].

2.1 Description of the SFIL process

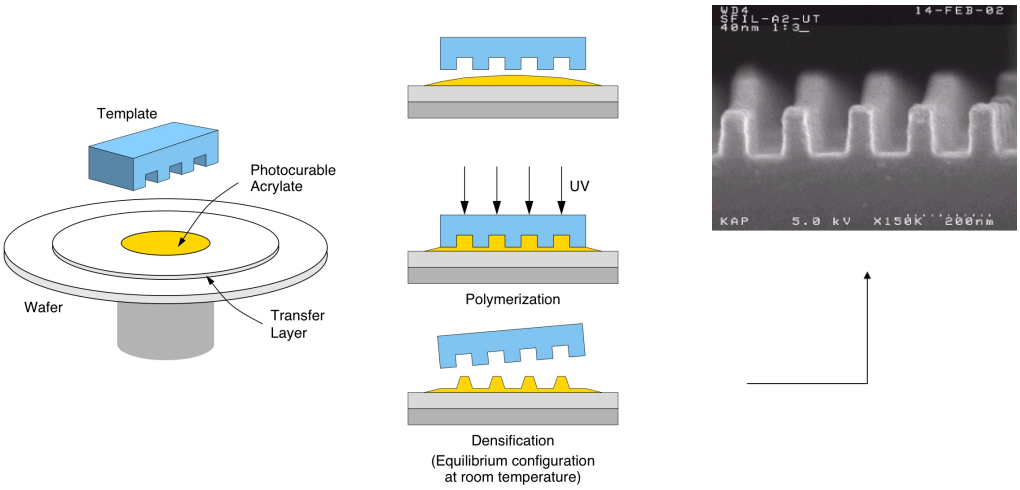


Figure 2.1: (a) A diagram of the SFIL process (not to scale) and (b) resulting patterns of size 40nm as seen using a scanning electron microscope.

Multiple separate processes have to be carried out to complete the

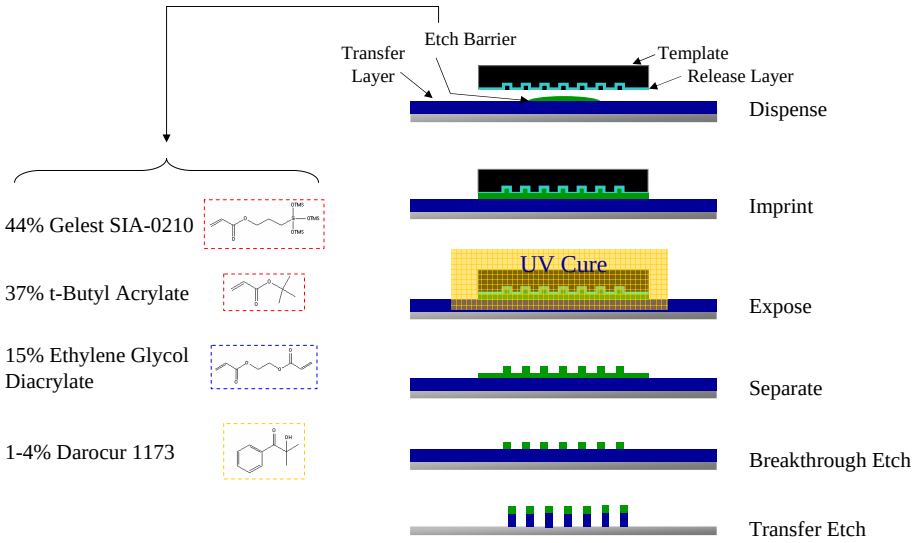


Figure 2.2: Constituents of the etch barrier solution and SFIL process flow.

pattern transfer (Figure 2.2). To create a pattern layer, an organic polymer layer (transfer layer) is spin-coated on a silicon substrate. A low viscosity, photopolymerizable, organosilicon solution (etch barrier) is then distributed on the wafer. A transparent template, which has patterned relief structures, is placed over the coated silicon substrate. This displaces the etch barrier solution which gets trapped in the pattern.

Once the pieces are in place, irradiation with UV light through the backside of the template cures the etch barrier into a cross-linked polymer film. A fluorocarbon release layer on the template allows separation from the substrate, leaving an organosilicon relief image that is a replica of the template pattern. A halogen etch is used to break through the undisplaced etch barrier material (residual layer) exposing the underlying transfer layer [10].

The advantages of SFIL are that it does not use projection optics but relies on photopolymerization chemistry and mechanical processes at room-temperature and low pressure. Unlike other imprint lithography techniques, it uses a low viscosity, photo-curable, organosilicon liquid and avoids high temperatures and imprint pressures. The transparent rigid imprint template allows flood exposure of the photopolymer to achieve cure, and enables classical optical techniques for layer-to-layer alignment.

2.2 Modeling polymerization in SFIL

We briefly describe the existing model of formation of polymeric lattices in SFIL. The details have been published elsewhere [61, 14, 15]. Free radical polymerization of the etch barrier solution results in a glassy polymer structure with a shape dictated by the patterns on the template. A mathematical polymerization model provides the topology of the polymer chains that form the etch barrier.

2.2.1 Constituents of the etch barrier solution

We model the chemical reactions between the constituents of the etch barrier solution when they are exposed to UV light. As seen in Figure 2.2, the solution contains four different chemical compounds in different weight proportions [42, 41] . Gelest SIA-0210 is a silicon containing monoacrylate for etch resistance. We refer to it as "monomer 1". It comprises 44% of the solution. t-Butyl Acrylate ("monomer 2") makes up for 37% of the solution

and is a reactive diluent to maintain low viscosity. Ethylene Glycol Diacrylate is the cross-linker used for mechanical stability. It takes up 15% of the weight of the solution. Darocur 1173 is the photoinitiator that forms free radicals on exposure to UV light. It accounts for 1-4% of the solution.

2.2.2 Polymerization reactions

The polymers are formed by Radical Polymerization. There are three main stages of such a reaction – initiation, chain propagation and chain termination. In the initiation stage, exposing the photoinitiator to UV light splits it into two free radicals. In chain propagation, the radicals react with other monomers or already radicalized polymer chains. This step leads to the large chains in polymers. Termination occurs when two chains that have free radicals react to form a single chain without a free radical.

Figure 2.3 shows the events that take place in free radical polymerization. We represent the photoinitiator by I, the monomers by M, the radicals by $R^\bullet$, radicalized polymers by $P^\bullet$, and the polymer chains formed by P.

$$
\begin{array}{rcll}
I & \xrightarrow{h\nu} & R^\bullet + R^\bullet & \left.\vphantom{\begin{array}{c}a\\a\end{array}}\right\} \text{Initiation} \\
R^\bullet + M & \rightarrow & P^\bullet & \left.\vphantom{\begin{array}{c}a\\a\end{array}}\right\} \text{Propagation} \\
P^\bullet + M & \rightarrow & P^\bullet & \\
P^\bullet + P^\bullet & \rightarrow & P & \left.\vphantom{\begin{array}{c}a\\a\end{array}}\right\} \text{Termination by combination}
\end{array}
$$

Figure 2.3: Reactions in free radical polymerization

2.2.3 Monte Carlo simulation of polymerization

A kinetic Monte Carlo algorithm simulates the chemical reactions that take place in the etch barrier solution [61, 27] . The inputs to the Monte Carlo algorithm are various reaction rates in form of probabilities and the ratio of individual constituents [42, 41] . Its output is a lattice-like stochastic cuboidal topology of monomers, cross-linker, initiator, substrate, and template molecules interacting with each other by central pair potentials. Figure 2.4 shows the various steps that take place on randomly chosen lattice sites. Empty lattice sites, also called voids, are introduced to facilitate motion of molecules. There is good agreement in the statistics of degree of polymerization between the simulations and experimentally observed quantities [57].

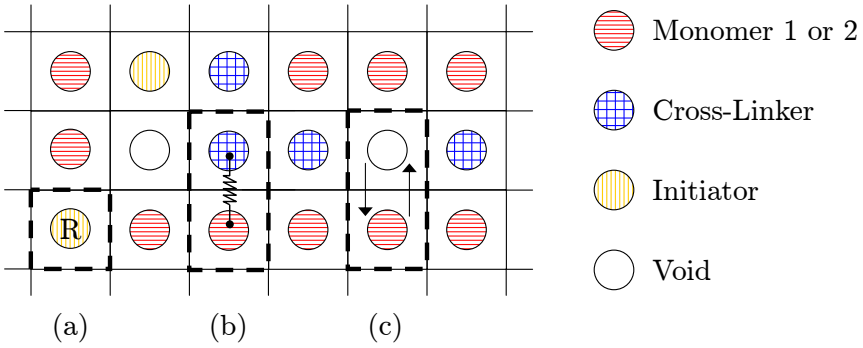


Figure 2.4: Steps of the Monte Carlo algorithm for polymer topology generation are shown on a part of the lattice. (a) An initiator can change to a radical. (b) A bond can form between two monomers or cross-linkers. (c) A molecule can move to an empty lattice site.

2.3 Modeling densification in SFIL

Polymerization is accompanied by densification due to the change in interaction potential between photopolymer precursors from Van der Waals to covalent. Typical change in the feature volume is around 9% [30, 27]. Densification affects the shape of the resulting structure. It is a very slow process compared to the reaction. Hence, polymerization is modeled separately from the subsequent densification.

2.3.1 Molecular statics base model

The result of the Monte Carlo algorithm is just a topology of molecules connected with different bonds. The molecules are treated as point masses in this model. We still do not know an equilibrium position of this network. Because of changes in bonds and the equilibrium lengths, pre-strain is built-in to the problem. The equilibrium configuration with Dirichlet boundary condition at the base (substrate) is found by letting the molecules relax to an equilibrium configuration [72].

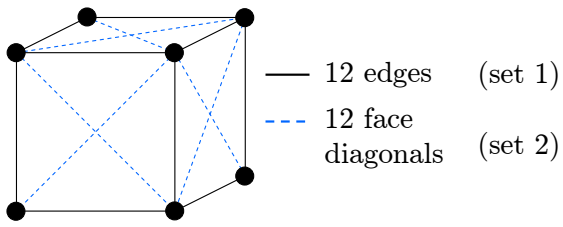


Figure 2.5: A single cell of the lattice showing edge and face bonds.

The molecules are connected to nearest neighbors along edges and faces

of the cubical lattice (Figure 2.5). Each molecule, unless it is on the boundary of the topology, is connected to 18 other molecules (assuming no "void" molecules as neighbors). The face diagonal bonds simulate volume exclusion and provide shear stiffness to the lattice. All of them are modeled by Lennard-Jones potential. Edge bonds can be both covalent and non-covalent (Van der Waals bonds). The Monte Carlo step provides this information. Covalent bonds are modeled by a stiff harmonic potential.

2.3.2 Lattice equilibrium equation

Consider a cubical lattice with N_1, N_2, and N_3 molecules in x, y, and z directions respectively. Let $\mathcal{A}$ be the set of all molecules, and

$$\boldsymbol{x} := \{\{\{x^{ijk}\}_{i=0}^{N_1-1}\}_{j=0}^{N_2-1}\}_{k=0}^{N_3-1}$$

be the set of all degrees of freedom (DOFs, coordinates of all the molecules). Similarly, $\boldsymbol{f}$ is the set of point forces on all molecules and $\bar{\boldsymbol{x}}$ is the set of the initial guesses of the DOFs. Let $\mathcal{D}$ be the set of particles and directions with Dirichlet boundary conditions.

$$\mathcal{D} := \{(i,j,k,p) : x_p^{ijk} = \bar{x}_p^{ijk} \text{ is fixed in direction } p\}$$

We can define the total potential energy J for this lattice as a function of $\boldsymbol{x}$.

$$\begin{aligned} J(\boldsymbol{x}) := \frac{1}{2} \sum_{x^{ijk} \in \boldsymbol{x}} \sum_{\substack{x^{lmn} \in \boldsymbol{x} \\ x^{lmn} \rightsquigarrow x^{ijk}}} E\left(\left|\left|x^{lmn} - x^{ijk}\right|\right|; (i,j,k), (l,m,n)\right) \\ - \sum_{x^{ijk} \in x} \sum_{\substack{p=1 \\ (i,j,k,p) \notin \mathcal{D}}}^{3} f_p^{ijk} x_p^{ijk} \end{aligned} \tag{2.1}$$

The symbol $\leftrightsquigarrow$ means "is connected to". Here E denotes a central bond potential function that depends on the distance between any two given molecules and the bond parameters. For a lattice with covalent bonds forming the polymer chains, the bond parameters depend on the location of the molecules in the lattice. For the sake of a simple expression for J, this dependence of bond parameters is hidden in the expression for E by making E a function that also depends on the lattice indices (i, j, k) and (l, m, n) . The factor of half takes care of double counting of each bond in the summation over neighbors.

The problem is to find an equilibrium position by minimizing the energy.

$$\text{Minimize } J(\boldsymbol{x}) : x_p^{ijk} = \bar{x}_p^{ijk} \ \ \forall\, (i, j, k, p) \in \mathcal{D}.$$

We use a Newton trust-region method implemented in TAO/PETSc [11, 20] to reach a local minimum starting from a given initial guess $\bar{\boldsymbol{x}}$. This requires derivatives of J. The first derivative is

$$J'|_{\boldsymbol{x}} = \left\{ \left. \frac{\partial J}{\partial x_p^{ijk}} \right|_{\boldsymbol{x}} \right\}_{i,j,k,p} = \boldsymbol{g} - \boldsymbol{f}$$

where

$$g_p^{ijk} = \sum_{\substack{x^{lmn} \in x \\ x^{lmn} \leftrightsquigarrow x^{ijk}}} \frac{\partial E}{\partial x_p^{ijk}} \left(\left|\left| x^{lmn} - x^{ijk} \right|\right| ; (i, j, k) , (l, m, n) \right) \text{ if } (i, j, k, p) \notin \mathcal{D}$$

and 0 otherwise. The vector $\boldsymbol{g}$ represents the pre-strain. The second derivative

is

$$J''|_{\boldsymbol{x}} = \left\{ \left\{ \left. \frac{\partial^2 J}{\partial x_p^{ijk} \partial x_q^{lmn}} \right|_{\boldsymbol{x}} \right\}_{i,j,k,p} \right\}_{l,m,n,q} =: H$$

where

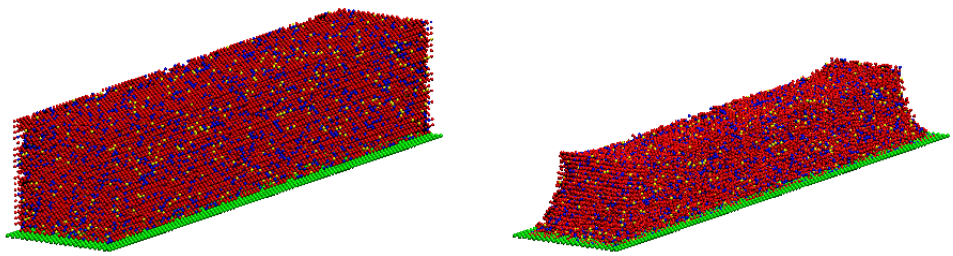
$$H_{p,q}^{ijk,lmn} = \frac{\partial^2 E}{\partial x_p^{ijk} \partial x_q^{lmn}} \left(\left|\left| x^{lmn} - x^{ijk} \right|\right| ; (i,j,k), (l,m,n) \right)$$

if $(i,j,k,p) \notin \mathcal{D}$ and $(l,m,n,q) \notin \mathcal{D}$. Otherwise the matrix entries are 0.

2.4 Numerical solution of the base model

By the "base model" we mean the molecular statics model that uses the exact information about the lattice topology. Numerical solution of such a molecular statics base model, which is assumed to describe the microstructure completely, is computationally very expensive. This is due to a large problem size, on the order of millions of DOFs. Rapid variation in material properties, ill-conditioning, nonlinearity, and existence of multiple minima make this problem even more challenging to solve.

Figure 2.6 shows a typical solution geometry visualized in VMD [52] with the pre-strained fine-scale lattice and the lattice in equilibrium. The biggest problem run so far had 3 millions DOFs (for a 100-cubed lattice). The computation of equilibrium took 370 CPU hours (divided amongst 64 processors) and 25000 Newton iterations [14].



(a) **(b)**

Figure 2.6: Part of the etch barrier modeled as a lattice of size $21 \times 101 \times 21$ in (a) pre-strained state and (b) equilibrium with fixed bottom layer. The colors correspond to different constituent molecules as shown in Figure 2.2. The solution was computed by Bauman [14].

2.5 Parameter estimation of molecular potentials

In this section we describe an inverse problem approach to determine the bond stiffness parameters of the molecular lattice from stress-strain experiments on poly(methyl methacrylate) (PMMA).

2.5.1 Experimental stress-strain relationship

PMMA, sold by trade name Plexiglas amongst others, is easily available in any desired form. The main reason why it is used to compute parameters for the SFIL lattice is that the monomer unit is an acrylate except for two methyl groups. Acrylates form the principle components of the compounds used to produce the etch barrier [41], Figure 2.7.

The PMMA sample is a small dog-bone shaped object with length 1.25 cm and width 0.4 cm (Figure 2.8). The sample is pulled at both ends

by forces such that the strain increases by 0.25% at each step. The maximum strain is 2.25% (Table 2.5.1). The experiments were conducted by Elizabeth Collister from the Willson Research Group [31].

Force (kN)	0.35	0.65	1.1	1.35	1.75	1.9	2.2	2.3	2.45
Stress (MPa)	7	13	22	27	35	38	44	46	49
Axial strain (%)	0.25	0.5	0.75	1	1.25	1.5	1.75	2	2.25

Table 2.1: Experimental data from the stress-strain experiments on PMMA.

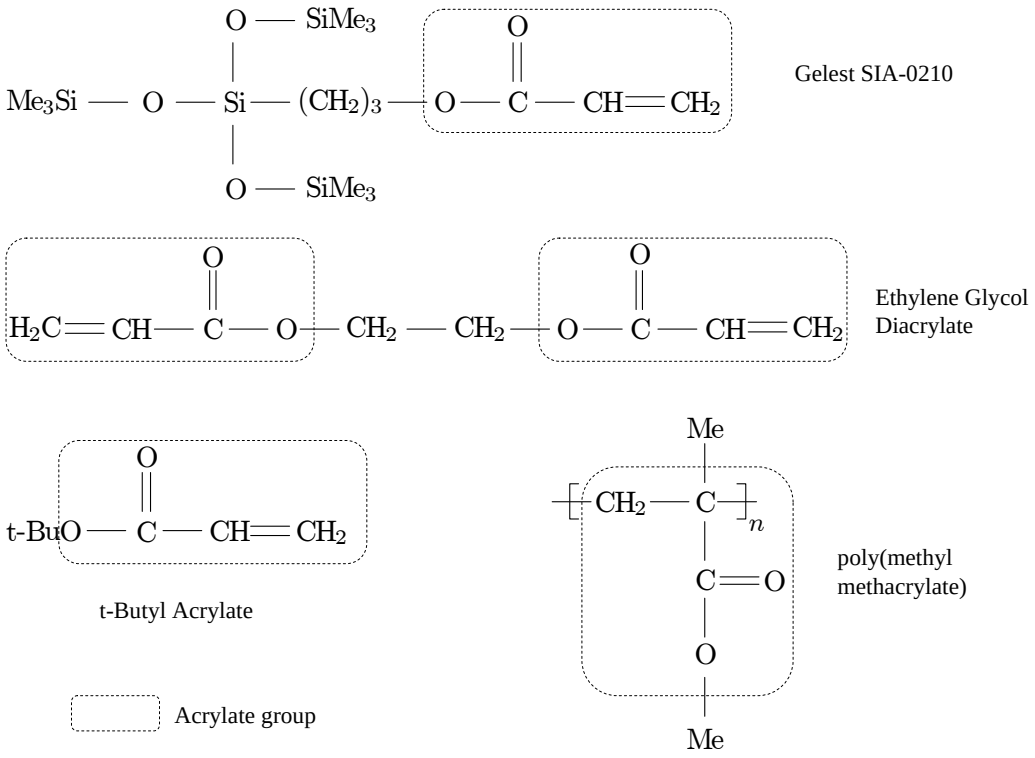


Figure 2.7: Molecular structure of three constituents of the etch barrier (Figure 2.2) and PMMA. Acrylate groups are marked [41].

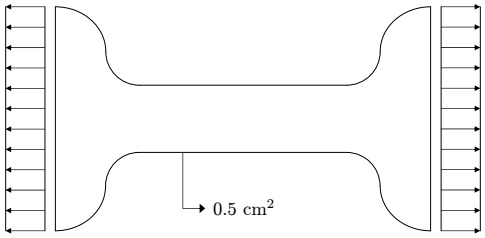


Figure 2.8: PMMA sample used for the stress-strain experiments.

2.5.2 Formulation of the inverse problem

We determine the spring potential parameters for a cubical lattice by using the data from the experimental stress-strain curve.

We use the superscript i to mark the i^{th} experiment. The forces used are $\{F^i\}_{i=1}^{N_E=9}$. To keep the notation simple, we use the superscript 0 for the quantities when no force is applied. Thus, $i = 0, 1, 2, \ldots, N_E$ and $F^0 = 0$.

We take a cubical lattice with N_1, N_2, and N_3 molecules in x, y, and z directions respectively. $\mathcal{A}$ is the set of all molecules, and

$$\boldsymbol{x} := \{\{\{x^{lmn}\}_{l=0}^{N_1-1}\}_{m=0}^{N_2-1}\}_{n=0}^{N_3-1}$$

is the set of all molecular coordinates. Let $\boldsymbol{x}^i$ denote the molecular coordinates when force F^i is used on the molecular lattice. We use the vector $\boldsymbol{p}$ to denote the unknown lattice potential parameters. Thus, $\boldsymbol{x}^i = \boldsymbol{x}^i(\boldsymbol{p})$.

Let $\mathcal{L}$ represent the length of the lattice in the direction of the force. $\mathcal{L}$ is a linear functional of $\boldsymbol{x}$ defined using the average distance between the cube surfaces perpendicular to direction of force. If the force is in x direction,

$$\mathcal{L}(\boldsymbol{x}) = \frac{1}{N_2 N_3} \sum_{m=0}^{N_2-1} \sum_{n=0}^{N_3-1} \left(x^{(N_1-1)mn} - x^{(0)mn} \right). \tag{2.2}$$

The observed quantities are the strains ϵ^i. In terms of the lengths, the observables are

$$\epsilon^i = \frac{\mathcal{L}(\boldsymbol{x}^i) - \mathcal{L}(\boldsymbol{x}^0)}{\mathcal{L}(\boldsymbol{x}^0)}.$$

Let q^i be the experimentally observed strains. The misfit as a function of the unknown parameters $\boldsymbol{p}$ is defined as

$$\mathcal{Q}(\boldsymbol{p}) = \frac{1}{2}\sum_{i=1}^{N_E}\left(\epsilon^i - q^i\right)^2.$$

Expanding the expression, we get

$$\mathcal{Q}(\boldsymbol{p}) = \frac{1}{2}\sum_{i=1}^{N_E}\left(\frac{\mathcal{L}(\boldsymbol{x}^i(\boldsymbol{p})) - \mathcal{L}(\boldsymbol{x}^0(\boldsymbol{p}))}{\mathcal{L}(\boldsymbol{x}^0(\boldsymbol{p}))} - q^i\right)^2.$$

We want to minimize the misfit as a function of $\boldsymbol{p}$. Looking ahead, to compute $\frac{\partial \mathcal{Q}}{\partial \boldsymbol{p}}$, we need $\frac{\partial \mathcal{L}}{\partial \boldsymbol{x}^i}$ and $\frac{\partial \boldsymbol{x}^i}{\partial \boldsymbol{p}}$. We can compute the first quantity from the definition of $\mathcal{L}$ in terms of $\boldsymbol{x}$ (Equation (2.2)). To compute the second quantity, we need to know the equilibrium equation for the lattice. Let $J = J(\boldsymbol{x}, \boldsymbol{p})$ be the total potential energy (Equation (2.1)) . At the minimum, $\frac{\partial J}{\partial \boldsymbol{x}} = 0 \ \ \forall \boldsymbol{p}$.

Assuming sufficient regularity, we differentiate this constraint with respect to $\boldsymbol{p}$. Thus,

$$\frac{\partial^2 J}{\partial \boldsymbol{x}^2}\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{p}} + \frac{\partial^2 J}{\partial \boldsymbol{x}\partial \boldsymbol{p}} = 0.$$

We assume $\frac{\partial^2 J}{\partial \boldsymbol{x}^2}$ is positive-definite, which implies that the equilibrium configuration $\boldsymbol{x}$ is a local minimum. Thus we can solve this linear system of equations

for $\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{p}}$. Finally, the gradient of $\mathcal{Q}$ can be computed as

$$\frac{\partial \mathcal{Q}}{\partial \boldsymbol{p}} = \sum_{i=1}^{N_E} \frac{\epsilon^i - q^i}{\mathcal{L}(\boldsymbol{x}^0)^2} \left[\left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{x}} \left(\frac{\partial^2 J}{\partial \boldsymbol{x}^2} \right)^{-1} \frac{\partial^2 J}{\partial \boldsymbol{x} \partial \boldsymbol{p}} \right) \Bigg|_{\boldsymbol{x}^0} \mathcal{L}(\boldsymbol{x}^i) \right.$$
$$\left. - \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{x}} \left(\frac{\partial^2 J}{\partial \boldsymbol{x}^2} \right)^{-1} \frac{\partial^2 J}{\partial \boldsymbol{x} \partial \boldsymbol{p}} \right) \Bigg|_{\boldsymbol{x}^i} \mathcal{L}(\boldsymbol{x}^0) \right].$$

All the quantities on the right hand side are computable. We use the steepest descent method to get the optimum $\boldsymbol{p}$.

2.5.3 Numerical results

We scaled the sample to match the area of a lattice with 51 cells on each size. We determined two spring constants for a lattice with harmonic springs for the covalent bond and the non-covalent bonds. Hence, $\boldsymbol{p} = (k_1, k_2)$. For this lattice, the optimum values were 1320 N/m for the covalent bond and 1085 N/m for the non-covalent bond. The experimental and lattice stress-strain data-points are shown in Figure 2.9.

2.6 Remarks on the inverse problem approach

The numerical results show that the bond parameters determined via the inverse problem approach result in a stress-strain relationship that matches well the experimentally determined stress-strain relationship. However, this does not imply that the optimal bond parameters *themselves* are accurate. This is because the experiments are carried out at length scale of centimeters whereas the size of polymerized material is on a length scale less than mi-

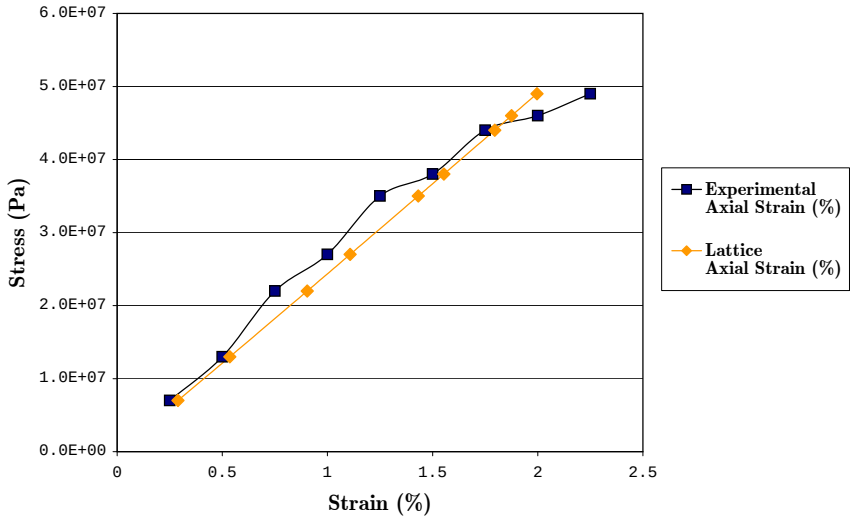


Figure 2.9: Experimental stress-strain curve for a PMMA sample and the best stress-strain curve for a lattice with 52 points in each side.

crometers. This is a physics based reason behind the possible inaccuracy. The second reason, a mathematical one, is that the inputs to the inverse problem are averaged forces and resulting strains (over the measurement surface). This makes any inference of the microstructure inherently ill-conditioned.

To clarify using a simple 1-D example, consider two linear springs (of possibly different stiffness) joined in series. It is impossible to determine the two individual stiffness constants by applying forces on the end-points only and measuring the relative displacement of the end-points. One can only determine a quantity that is a function of the harmonic average. Thus, it would be best to get the values of the bond parameters from an *ab initio* approach. This is a challenging problem by itself and outside the scope of this dissertation.

We also note that the inverse problem gives deterministic values of

the bond parameters. The lattice, however, is stochastic in nature since it is generated by a Monte Carlo simulation of the chemical reactions. Thus, one may want to quantify this uncertainty using a probability distribution function for unknown parameters. This requires multiple solution of the inverse problem for a brute-force determination. We do not consider the stochastic case.

Chapter 3

Goal-oriented *hp*-adaptivity for Dimensional Reduction

hp-adaptivity is a widely applicable mesh adaptivity technique for computing approximate solutions of partial differential equations [38, 37, 39] . This chapter describes an application of the *hp*-adaptivity algorithm for lattice problems to achieve dimensional reduction of the problem space for an approximate solution. The adaptivity algorithm chooses the mesh on top of the lattice so that large elements with possibly high polynomial degree are chosen in regions with mild variation in the solution. Elsewhere the mesh resolves the rapid variation by *h*-refinements and it resembles the underlying lattice. Both goal-oriented and energy-oriented local error estimates can be used to drive the mesh adaptivity [53] .

Our dimensional reduction approach is similar to the Quasicontinuum Method [62] and can be treated as its generalization. Instead of assuming a uniform deformation gradient (via the Cauchy-Born rule, see [46]), we use higher order polynomial interpolation within an element to kinematically constrain the masses and reduce the effective number of degrees of freedom.

3.1 Dimensional reduction

Let $\mathcal{X} := \mathbb{R}^N$ be the space of lattice DOFs. Let $\mathcal{V} \subset \mathcal{X}$ be the space of test vectors. The "stiffness matrix" $K \in \mathbb{R}^{N\times N}$ is symmetric and assumed positive definite on $\mathcal{V}$. $u_D \in \mathcal{X}$ is the Dirichlet boundary condition. Vector $f \in \mathcal{X}$ is the load. With this notation, the abstract variational formulation of the lattice elasticity problem is

$$\text{Find } u \in u_D + \mathcal{V} \text{ such that } v^T K u = v^T f \quad \forall\, v \in \mathcal{V}. \tag{3.1}$$

The solution u can also be characterized as the element of $\mathcal{X}$ that minimizes the energy functional J.

$$u = \arg\min J(v) \text{ where } v \in u_D + \mathcal{V} \text{ and } J(v) := \frac{1}{2} v^T K v - v^T f \tag{3.2}$$

In the spirit of Finite Element Methods, we approximate the exact solution by minimizing the energy in a subspace of $\mathcal{X}$. Let $\mathcal{X}_{hp}$ be a subspace of $\mathcal{X}$ and $\mathcal{V}_{hp}$ a subspace of $\mathcal{V}$. The vector dimension of $\mathcal{X}_{hp}$ is M, such that $M \leq N$. The subspaces chosen are such that the Dirichlet boundary condition u_D can be imposed exactly. Let A be a global interpolation operator. $A : \mathcal{X}_{hp} \to \mathcal{X}$. Hence, as a matrix, $A \in \mathbb{R}^{N\times M}$. Using the definition of A, the dimensionally reduced solution is

$$u_{hp} = \arg\min \left(J(Av_{hp}) = \frac{1}{2}(Av_{hp})^T K (Av_{hp}) - (Av_{hp})^T f \right) \tag{3.3}$$

where $hp \in u_D + \mathcal{V}_{hp}$. It is readily seen that minimizing the energy in a subspace means working with an effective stiffness matrix $A^T K A =: \widehat{K} \in \mathbb{S}^M_+$

and an effective load $A^T f =: \widehat{f} \in \mathbb{R}^M$. Whether these effective properties lead to an accurate approximation depends on the boundary conditions, spatial variations in K and f, and the choice of A.

Once the subspace $\mathcal{X}_{hp}$ and the interpolation operator A are chosen, the solution of Equation (3.3) is the best approximation of the solution of the original fine-scale problem (Equation (3.2)) in the energy norm.

3.2 Dimensional reduction using *hp*-adaptivity

We use the automatic *hp*-adaptivity algorithm [37, 39] to construct and automatically adapt the subspace $\mathcal{X}_{hp}$. For simplicity, we explain *hp* meshes for a lattice in 1-D. Groups of adjacent lattice cells form a single element. The number of degrees of freedom in each original element gives the element size h. Numerically h is the number of DOFs -1. The DOFs are constrained to lie on a local polynomial of degree p. The goal of the *hp*-adaptivity algorithm is to choose such a mesh automatically and optimally.

Consider a part of a 1-D lattice consisting of two adjacent elements in Figure 3.1. For the element on left, we have used a $p = 1$ polynomial to effectively constrain the node in the middle. For the element on right, a $p = 2$ polynomial reduces the element DOFs from 5 to 3. The shape functions to define the local polynomials are typical higher-order finite element shape functions.

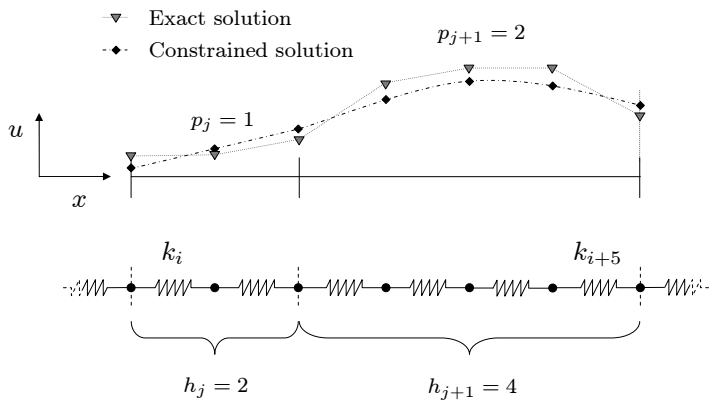


Figure 3.1: A piecewise polynomial is used to approximate the exact solution on two elements.

3.3 Goal-oriented *hp*-adaptivity for elliptic problems

The approximation strategy presented in the previous section produces solutions that have the least error measured in the energy norm. However, many times we are interested in only a linear functional of the full solution [39]. For example, in case of the lattice, we may be interested in the average deformation of an edge. In such a case, local error estimates to refine the mesh should reflect the error in the goal and not the energy. We describe this modification now.

We work with abstract variational forms in this section. Let the goal $\mathcal{G} : \mathcal{X} \to \mathbb{R}$ be a linear and continuous functional of the solution. Functional $\mathcal{G}$ can be characterized by an element g in $\mathcal{X}$. $\mathcal{G}(u) := (g, u)$ where $(\cdot, \cdot)$ is the canonical inner product on $\mathcal{X}$. Let $\mathcal{B} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a continuous, coercive bilinear form and $\mathcal{L} : \mathcal{X} \to \mathbb{R}$ be a continuous linear form. We keep track of the interpolation operator $A : \mathcal{X}_{hp} \to \mathcal{X}$ also. This is done to compare the

error estimates obtained here with those obtained later in Section 4.10.

The exact primal problem is

$$\text{Find } u \in u_D + \mathcal{V} \text{ such that } \mathcal{B}(u, v) = \mathcal{L}(v) \quad \forall\, v \in \mathcal{V}.$$

The approximate primal problem is

$$\text{Find } u_{hp} \in u_D + \mathcal{V}_{hp} \text{ such that } \mathcal{B}(Au_{hp}, Av_{hp}) = \mathcal{L}(Av_{hp}) \quad \forall\, v_{hp} \in \mathcal{V}_{hp}.$$

Let $e := u - Au_{hp}$ be the error due to this approximation. Let $r \in \mathcal{X}'$ be the residual.

$$r(v) = \mathcal{B}(e, v) \quad \forall\, v \in \mathcal{X}.$$

Using Galerkin orthogonality, $\mathcal{B}(e, Av_{hp}) = 0 \ \forall\, v \in \mathcal{V}_{hp}$. Hence, $r(v_{hp}) = 0 \ \forall\, v \in \mathcal{V}_{hp}$.

Let $w : \mathcal{X}' \to \mathbb{R}$, $w(r) := \mathcal{G}(e)$. Functional w is called the influence function. By reflexivity of $\mathcal{X}$, $r(w) = \mathcal{G}(e)$. Hence, $\mathcal{B}(e, w) = \mathcal{G}(e) \quad \forall\, e \in \mathcal{X}$. Since $\mathcal{B}$ is self-adjoint, the variational form for the exact error in the goal is

$$\text{Find } w \in \mathcal{V} \text{ such that } \mathcal{B}(w, e) = \mathcal{G}(e) \quad \forall\, e \in \mathcal{V}.$$

This is also known as the adjoint problem. The approximation of this adjoint problem is

$$\text{Find } w_{hp} \in \mathcal{V}_{hp} \text{ such that } \mathcal{B}(Aw_{hp}, Ae_{hp}) = \mathcal{G}(Ae_{hp}) \quad \forall\, e_{hp} \in \mathcal{V}_{hp}.$$

We can express the exact error in the goal in terms of the primal and adjoint solutions. The error is $\mathcal{G}(u) - \mathcal{G}(Au_{hp}) = \mathcal{G}(u - Au_{hp})$. This is equal

to $\mathcal{B}(u - Au_{hp}, w)$ using the adjoint problem. If we use Galerkin orthogonality and subtract a term that is identically zero, we get $\mathcal{B}(u - Au_{hp}, w - Aw_{hp})$. We can split this expression into a sum over elements.

$$\mathcal{G}(u) - \mathcal{G}(Au_{hp}) \; = \sum_{\text{elements } j} \mathcal{B}_j(u - Au_{hp}, w - Aw_{hp})$$

$\mathcal{B}_j$ is the same as $\mathcal{B}$ restricted to the j^{th} element. Of course this expression is usable only if we already know the exact primal and adjoint solutions.

We use this idea of computing the element-wise error estimates in the following way. We work with two meshes, one coarse and one fine, where the fine mesh is a uniformly refined version of the coarse mesh. The solution on the fine mesh serves as a reference for the exact solution and it is used to compute element-wise error estimates on the coarse mesh. These estimates dictate which coarse mesh elements should actually be refined for the next coarse mesh.

Let the superscripts f and c denote the fine and coarse meshes respectively. Let u^f and u^c denote the fine and coarse mesh solutions projected to the full lattice using fine and coarse mesh interpolation operators A^f and A^c respectively. We estimate the error in the goal by $\mathcal{G}(u^f) - \mathcal{G}(u^c) = \mathcal{G}(u^f - u^c)$. Using the definition of the adjoint problem, this is equal to $\mathcal{B}(u^f - u^c, w^f)$. Let Π^c be the projection-based interpolation operator from $\mathcal{X}$ to the coarse mesh [36] . Using Π^c and Galerkin orthogonality, we can subtract a term that is identically zero. The error then becomes $\mathcal{B}(u^f - u^c, w^f - \Pi^c w^f)$. If we use the projection-based interpolation operator for the primal solution as well, the

error becomes

$$\begin{aligned}\mathcal{B}(u^f - u^c, w^f - \Pi^c w^f) &= \mathcal{B}(u^f - \Pi^c u^f, w^f - \Pi^c w^f) \\ &+ \mathcal{B}(\Pi^c u^f - u^c, w^f - \Pi^c w^f).\end{aligned}$$

The second term is ignored [1] . We end up with following error estimate divided element-wise.

$$\begin{aligned}\left|\mathcal{B}(u^f - \Pi^c w^f, w^f - \Pi^c w^f)\right| &= \left|\sum_{\text{elements } j} \mathcal{B}_j(u^f - \Pi^c u^f, w^f - \Pi^c w^f)\right| \\ &\leq \sum_{\text{elements } j} \left|\left|u^f - \Pi^c u^f\right|\right|_{\mathcal{B}_j} \left|\left|w^f - \Pi^c w^f\right|\right|_{\mathcal{B}_j}\end{aligned}$$

Computing these estimates requires solving the primal and dual problems on coarse and fine meshes. For comparison, the element-wise error estimate for the energy-oriented adaptivity looks as follows.

$$\left|\left|u^f - \Pi^c u^f\right|\right|^2_{\mathcal{B}} = \mathcal{B}(u^f - \Pi^c u^f, u^f - \Pi^c u^f) = \sum_{\text{elements } j} \left|\left|u^f - \Pi^c u^f\right|\right|^2_{\mathcal{B}_j}$$

3.4 Numerical results

We present numerical results for a 1-D linear spring model with harmonic springs and a 2-D linearized model. The 2-D model also shows results for the goal-oriented adaptivity.

[1] This assumption is based largely on existing numerical experimentation [39] and existing estimates for the h version of Finite Element Methods.

3.4.1 1-D model with harmonic springs

The lattice consists of 1024 harmonic springs with identical equilibrium lengths and varying stiffnesses. Dirichlet boundary conditions constrain both end-points, Figure 3.2. The lattice is divided into 4 equal regions. Spring stiffness varies smoothly within each region, Figure 3.3. Overlaid in the same figure is the final mesh with different polynomial orders in different elements. The adaptivity algorithm chooses small elements in regions of sudden variations and larger elements with higher p when the changes are gradual. As seen in Figure 3.4, just 22 DOFs are enough to make the solution visually identical to the exact solution with 1025 DOFs. Figure 3.5 shows the convergence history. The error reaches 2% with just 10% DOFs.

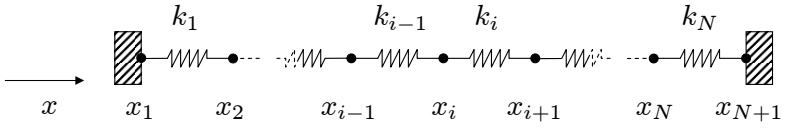


Figure 3.2: A one-dimensional spring network in static equilibrium. Both end-points are fixed by Dirichlet boundary conditions.

3.4.2 2-D linearized model

For a rectangular lattice in 2-D (Figure 3.6), a cell consists of four particles at the cell corners, connected by four shared edge springs and 2 diagonal springs. An element of the mesh consists of a rectangular patch consisting of multiple cells. We use tensor product polynomial spaces for the shape functions.

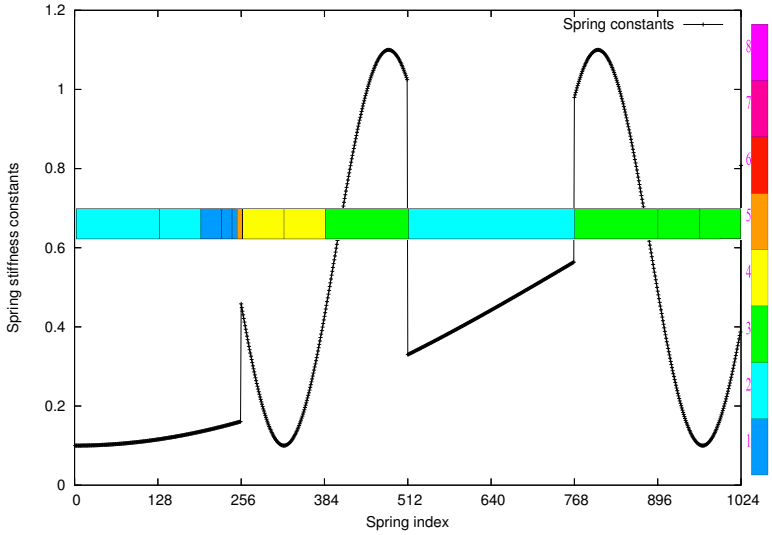


Figure 3.3: A piecewise smooth spring stiffness data shows that larger elements are selected in regions of slow variation. The color-coded polynomial scale is overlaid.

Because of geometric nonlinearity, the equations for equilibrium are not linear. In this example, we assume small strains and work with the linearized model only. Let $\mathcal{A}$ be the set of all particles in the lattice. Let $\mathbf{X}_\alpha, \alpha \in \mathcal{A}$ denote the position vector of particle α and $\mathbf{u}_\alpha$ be its displacement vector. Define $(\cdot)_{\alpha\beta} = (\cdot)_\alpha - (\cdot)_\beta$ for vectors. Let $k_{\alpha\beta}$ and $l_{\alpha\beta}$ be the stiffness and equilibrium length respectively of the spring connecting α and β. Define $\Delta l_{\alpha\beta} = ||\mathbf{X}_{\alpha\beta}|| - l_{\alpha\beta}$. Non-zero $\Delta l_{\alpha\beta}$ implies existence of pre-strain in the particular spring. Let $\mathbf{F}_\alpha^{EXT}$ be the external force on particle α. The linearized

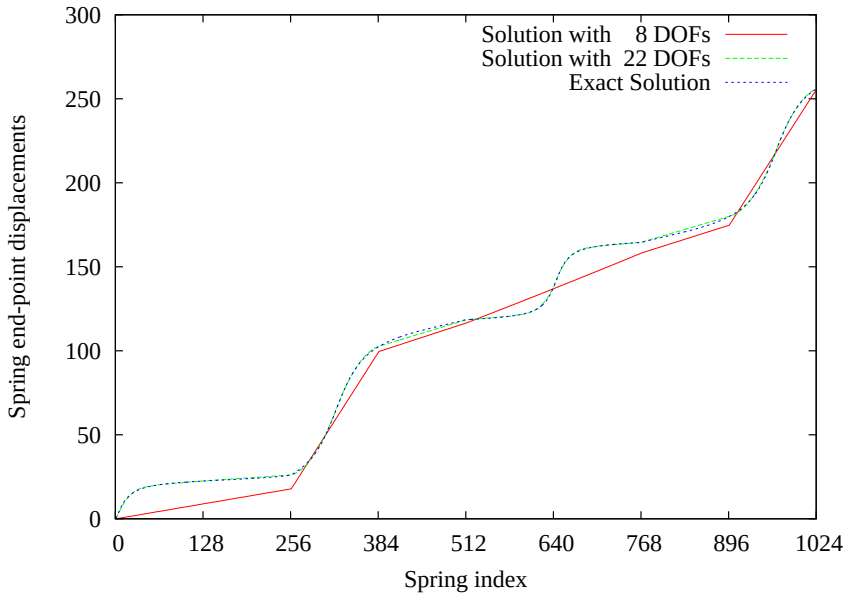


Figure 3.4: Convergence of solution for a 1025 DOF spring system with data shown in Figure 3.3.

variational form for finding the displacements $\{\mathbf{u}_\eta\}_{\eta\in\mathcal{A}}$ follows.

$$\begin{cases} \text{Find } \{\mathbf{u}_\eta\}_{\eta\in\mathcal{A}} \text{ such that Dirichlet boundary conditions are satisfied and} \\ \mathcal{B}\left(\{\mathbf{u}_\eta\}_{\eta\in\mathcal{A}}, \{\mathbf{v}_\zeta\}_{\zeta\in\mathcal{A}}\right) := \displaystyle\sum_{\substack{\alpha,\,\beta\in\mathcal{A} \\ \alpha\leftrightsquigarrow\beta}} \frac{1}{2}\frac{k_{\alpha\beta}}{||\mathbf{X}_{\alpha\beta}||^2}(\mathbf{X}_{\alpha\beta}\cdot\mathbf{u}_{\alpha\beta})(\mathbf{X}_{\alpha\beta}\cdot\mathbf{v}_{\alpha\beta}) \\ \qquad = \\ \mathcal{L}\left(\{\mathbf{v}_\zeta\}_{\zeta\in\mathcal{A}}\right) := \displaystyle\sum_{\alpha\in\mathcal{A}} \mathbf{F}_\alpha^{EXT}\cdot\mathbf{v}_\alpha \quad - \sum_{\substack{\alpha,\beta\in\mathcal{A} \\ \alpha\leftrightsquigarrow\beta}} \frac{1}{2}\frac{k_{\alpha\beta}\Delta l_{\alpha\beta}}{||\mathbf{X}_{\alpha\beta}||}(\mathbf{X}_{\alpha\beta}\cdot\mathbf{v}_{\alpha\beta}) \\ \forall \text{ admissible } \{\mathbf{v}_\zeta\}_{\zeta\in\mathcal{A}} \end{cases}$$

"$\alpha \leftrightsquigarrow \beta$" means particle α is connected to particle β (by the spring $(k_{\alpha\beta}, l_{\alpha\beta})$).

We chose a square lattice consisting of 64 cells in each direction, Fig-

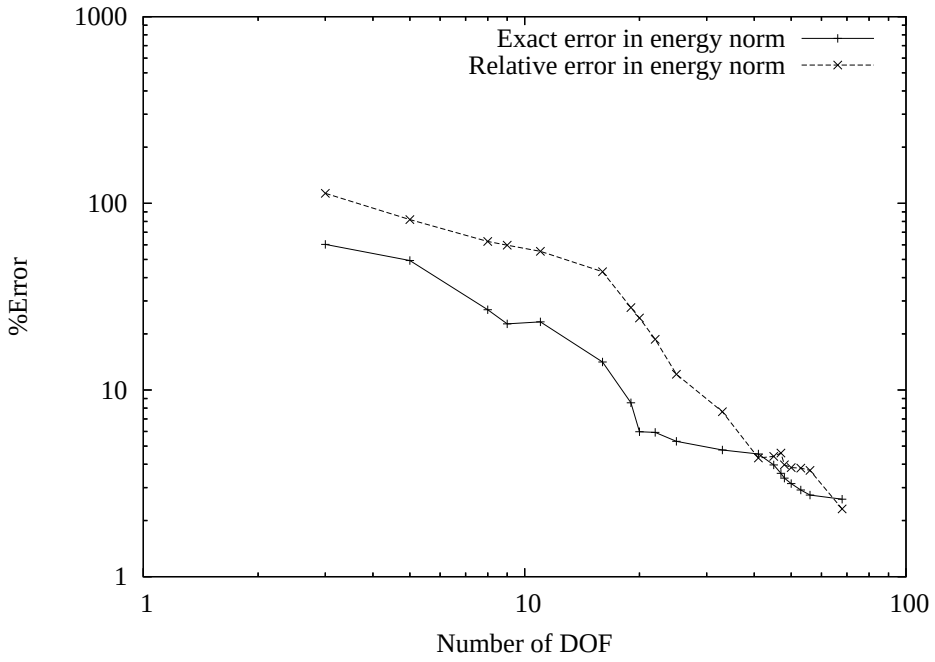


Figure 3.5: Convergence history for exact and relative errors in energy norm for energy-oriented *hp*-adaptivity for data shown in Figure 3.3.

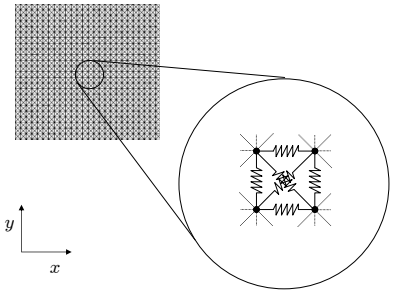


Figure 3.6: A 2-D rectangular lattice with a zoomed-in cell.

ure 3.7(a). The bottom was fixed and a unit force in x direction was applied at the top right corner. All spring constants were chosen to be 1. Edge springs had a length 1 and diagonal springs had a length $\sqrt{2}$. The equilibrium solution for this configuration is shown in Figure 3.7(b). For the goal-oriented

algorithm, the goal is defined to be the x displacement of the top-left corner. Figure 3.8 shows the hp-meshes using [35] with error less than 1% and number of DOFs almost 100 times fewer as compared to the fine-scale problem size. The corresponding exact and approximate solutions are visually indistinguishable at scale of the figures. Figure 3.9 shows the convergence histories of energy-oriented and goal-oriented adaptivity.

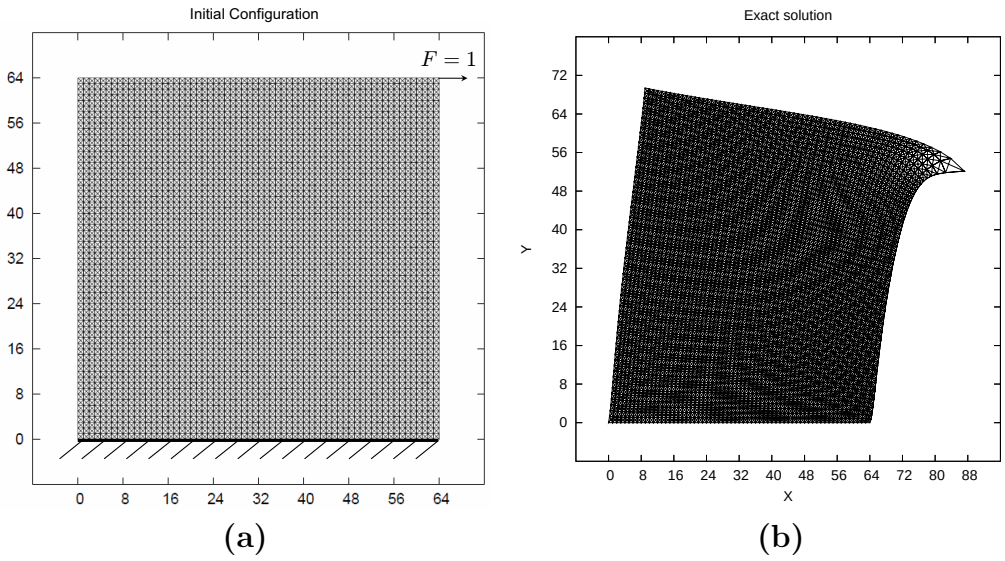
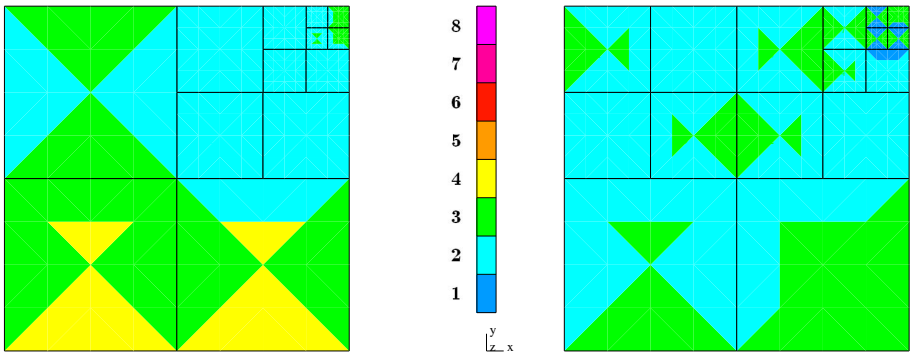


Figure 3.7: The 2-D lattice problem and its exact solution

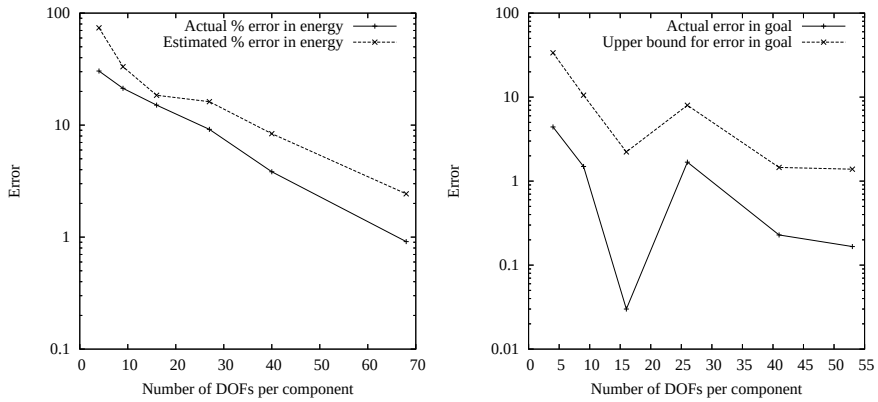
3.5 Dimensional reduction and homogenization

Numerical results of the previous section show that adaptive mesh refinement can be used for dimensional reduction of lattice based problems. It essentially amounts to solving the fine-scale model in a subspace. However,



(a) **(b)**

Figure 3.8: Final meshes for (a) energy-oriented and (b) goal-oriented *hp*-adaptivity.



(a) **(b)**

Figure 3.9: Convergence history for energy-oriented and goal-oriented *hp*-adaptivity.

if this approach is tried on a lattice model with fast varying "random" data, we see that the mesh refinement algorithm keeps on refining without reducing the error appreciably. Every time an element is refined, more fine-scales

features are noticed and they cannot be approximated by coarse-scale smooth functions. One of the meshes in the sequence of mesh refinements is shown in Figure 3.10.

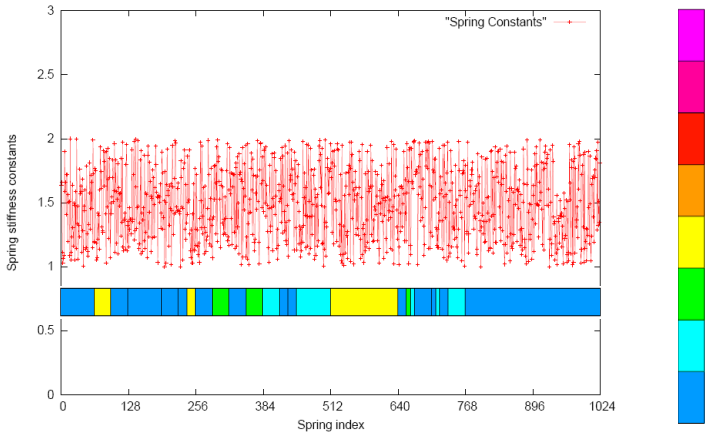


Figure 3.10: For rough material data, refinements happen everywhere without reducing the error appreciably. The mesh is overlaid on the 1024 random spring constants uniformly distributed in [1,2]. Compare Figure 3.3.

As we will also see in the next chapter, approximation in a subspace of smooth basis functions is insufficient if the material properties vary rapidly. In this chapter we have minimized the energy by averaging the stiffness matrix. As it is well known in the theory of homogenization of continuous periodic media, homogenization involves averaging of the inverse of the stiffness [49]. The next chapter uses this result as a motivation for developing an adaptive framework for homogenization.

Chapter 4

Local Numerical Homogenization

We describe a method for local numerical homogenization of nonlinear lattice elasticity. There is no assumption of periodicity of material properties or boundary conditions. The method will be applied to the problem of molecular statics encountered in SFIL (Chapter 2). However, the method is general and applicable for problems with a given fine mesh that sufficiently resolves the fine-scale material properties.

Traditionally, a finite element mesh is designed after obtaining material properties in different regions. The mesh has to match material discontinuities. In our approach we do exactly the opposite. A coarse mesh is selected first and homogenization is done on each individual element of the mesh. In this way, the coarse mesh and the fine-scale structure become naturally compatible. On each element, the homogenization method works with the Moore-Penrose pseudoinverse of the element stiffness matrix to produce pseudoinverse of the local homogenized stiffness matrix as the output. This output depends on the local load as well as a chosen local interpolation from the coarse to the fine mesh. This process requires solving a continuous-time Lyapunov equation on each element. The pseudoinverse of the homogenized stiffness matrix is

then pseudo-inverted to get the homogenized stiffness matrix for the assembly phase.

For computational efficiency of our method, we require fast and possibly approximate algorithms for Moore-Penrose pseudoinverse. This is presented in Chapter 5. Finally, we integrate the homogenization method with goal-oriented mesh adaptivity and Newton iterations for linearizing the nonlinear lattice elasticity problem. The numerical results are presented in Chapter 6.

4.1 Using interpolation for dimensional reduction

In Chapter 3, we extended *hp*-adaptivity to problems posed on lattices. The method computed local effective material properties in form of a local stiffness matrix. The approximate solutions matched the exact solutions very well. However, in general, the resulting effective properties are incorrect. The error becomes worse when the material properties have large variations. A simple example in 1-D shows the discrepancy.

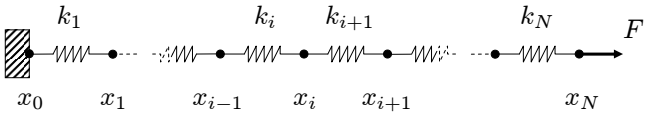


Figure 4.1: N springs with a fixed leftmost particle and a force F on the rightmost particle.

As shown in Figure 4.1, let us have N harmonic springs, $N \geq 2$, with identical equilibrium lengths and variable stiffnesses $\{k_i\}_{i=1}^{N}$. Assume the left end-point is fixed at origin and that a force F is applied on the right end-

point. It is readily seen that the tension in each spring is F and hence the displacement of the right end-point is F/k, where the effective stiffness k is the harmonic mean of $\{k_i\}_{i=1}^N$ times $1/N$.

$$\frac{1}{k} = \sum_{i=1}^{N} \frac{1}{k_i}. \tag{4.1}$$

For this problem, the displacements $\{u_i\}_{i=0}^N =: u$ can also be found by minimizing the total potential energy J.

$$\text{Minimize } J(u) = \frac{1}{2}\sum_{i=1}^{N} k_i(u_i - u_{i-1})^2 - Fu_N \quad \text{ such that } u_0 = 0. \tag{4.2}$$

We can minimize J in a subspace. Let us consider all the springs to form a single element and assume a linear constraint $u_i = (i/N)u_N$, where u_N is the only unknown. This interpolation scheme is independent of the specific values of k_i. The total potential energy in terms of a single unknown u_N is

$$\frac{u_N^2}{2N^2}\sum_{i=1}^{N} k_i - Fu_N.$$

The total energy is minimum when

$$u_N = \frac{N^2 F}{\sum_{i=1}^{N} k_i}.$$

This shows that the effective stiffness, given by F/u_N, is the arithmetic mean of $\{k_i\}_{i=1}^N$ times $1/N$.

$$k = \frac{1}{N^2}\sum_{i=1}^{N} k_i. \tag{4.3}$$

For positive quantities, the harmonic mean is always less than the arithmetic mean unless all the quantities are equal. This can be proved using induction on N. Hence, in 1-D, if we minimize in a subspace using the typical

finite element shape functions, we always overestimate the effective stiffness. The relative error due to this approximation is not bounded. For example, take $N = 2$ and $k_1 = 1$. In this case, the exact effective stiffness is always less than 1 but the arithmetic mean stiffness can be made larger than any arbitrary value by increasing k_2.

Although this 1-D problem is simple, its analysis clearly shows the pitfalls associated with homogenization. Problems in higher dimensions are more complex and there may not be a simple effective property like in 1-D. In fact, in the case presented above, the effective material property was just one of the many possible ones. The one chosen was to match approximate end-point displacement with the exact one. It does not tell us what happens as we move away from the end-point. Thus, in a general case, homogenized properties will also depends on volumetric forces and types of boundary conditions and their distribution. In addition, the choice of a coarse mesh plays an important role for the numerical homogenization.

We view this as a departure point and attempt local homogenization from a different point of view. Our emphasis is on computing best local material properties (best in a chosen norm or semi-norm) so that the upscaling averages the inverse of the stiffness matrix.

4.2 Local numerical homogenization

We change our focus from the best energy norm solution in a simple subspace. Our focus is on getting locally best effective material properties

from the linearized operator of the nonlinear lattice elasticity problem. For an iterative numerical scheme, the material properties are the element stiffness matrices that depend on the current coarse mesh and the current solution guess.

We begin by introducing a coarse grid compatible with the lattice, as illustrated in Figure 4.2. In each coarse element, the DOFs are displacements of the corners. We find the effective stiffness for each element (Sections 4.3 and 4.4) in which every coarse DOF interacts with each other. The local effective stiffness matrices are then assembled to form the global stiffness matrix and the problem is solved using the standard Finite Element technology. The DOFs corresponding to the hanging nodes are eliminated before the assembly stage.

Consider a lattice with a load provided by the built-in pre-strain, the known external loads, and unknown reactions. Without solving the fine-scale problem exactly we cannot know the forces on the masses forming a detached coarse element. However, given the location of the masses, we can compute the pre-strain in each element. To homogenize locally, we can use the pre-strain as a particular load for which we want accurate homogenization. We may also choose to homogenize without considering any particular load. We will formulate the homogenization method using both choices (separately). For the moment, we work with a known non-zero self-equilibrated load denoted by f, $f \in \mathbb{R}^N$. In Section 4.4 we consider the case $f = 0$ which can be understood as homogenization for all loads.

Let K be a symmetric element stiffness matrix of size N. The local

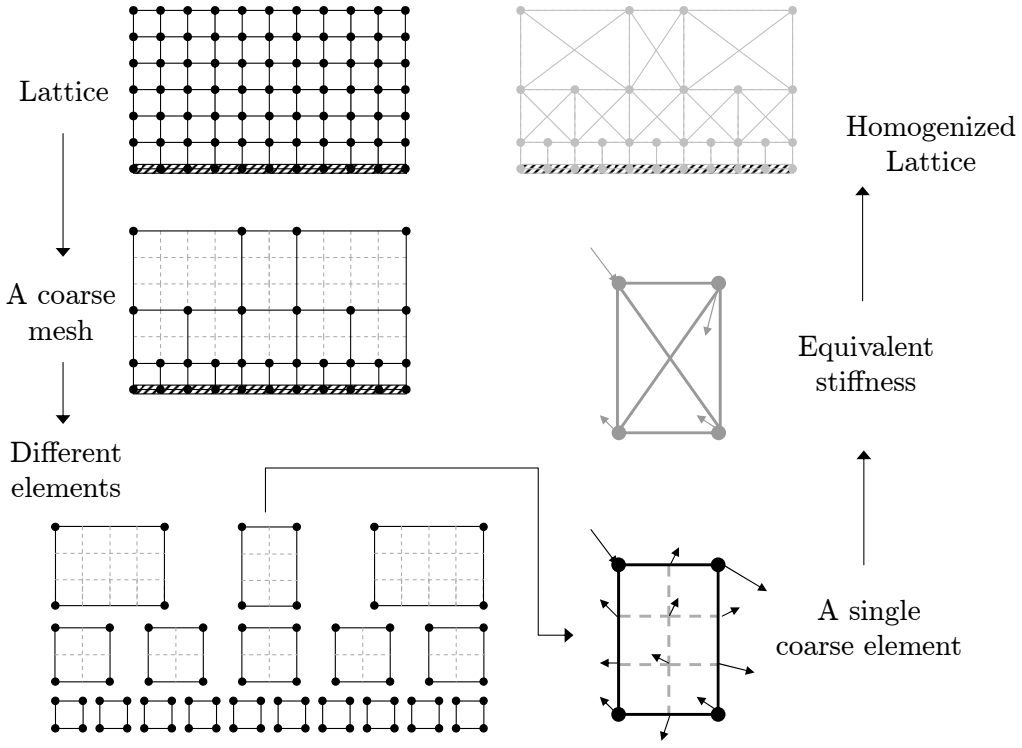


Figure 4.2: A 2-D lattice with fixed bottom layer is approximated with a coarse mesh. Effective local stiffness is computed on coarse elements and used to create a global effective lattice. The diagonal bonds in the fine lattice are not shown for clarity.

equation for $u \in \mathbb{R}^N$ is

$$Ku = f.$$

The exact solution is given by

$$u = K^\dagger f + u_0$$

where $K^\dagger$ is the Moore-Penrose pseudoinverse [24] of K and u_0 is an arbitrary vector in the null space $\mathcal{N}(K)$ of K.

We mark the coarse-scale quantities with a hat. For $M \leq N$, let $\widehat{K} \in \mathbb{R}^{M \times M}$, as yet unknown, represent the effective element stiffness for the coarse element DOFs. Let $\widehat{f} \in \mathbb{R}^M$ be the coarse-scale load. $A \in \mathbb{R}^{N \times M}$ is a chosen interpolation operator. It is local to the element. Vector $\widehat{f} = A^T f$ is the natural interpolation of the fine-scale load f.

In terms of the unknown $\widehat{K}$, the coarse-scale solution is

$$\widehat{u} = \widehat{K}^\dagger A^T f + \widehat{u_0}$$

where $\widehat{K}^\dagger$ is the Moore-Penrose pseudoinverse of $\widehat{K}$ and $\widehat{u_0} \in \mathcal{N}(\widehat{K})$. When compared to the fine-scale solution, the error is

$$u - A\widehat{u} = (K^\dagger - A\widehat{K}^\dagger A^T)f + (u_0 - A\widehat{u_0}).$$

Hence, up to an arbitrary constant, the error $e \in \mathbb{R}^N$ is

$$e := (K^\dagger - A\widehat{K}^\dagger A^T)f.$$

Let B be a symmetric positive-definite matrix of size N. We will use B to measure the error e and in process define a homogenization problem to compute $K^\dagger$.

B should not approximate energy norm. Suppose K is positive definite, and B is chosen to be K. Then we are minimizing the error in the energy norm in a round-about way, and as we have seen this results in an incorrect homogenization when the interpolation operator A is a material independent.

Section 4.3 defines the local homogenization problem for a given self-equilibrated load. If the load is not known, or is zero, we modify the local

homogenization procedure to compute best effective properties for arbitrary loads. This takes care of forces on an element due to other elements. Homogenization for all loads (by not using any given load) is explained in Section 4.4.

4.3 Locally best effective properties for a given load

We now define and solve the local homogenization problem for a given non-zero and self-equilibrated load f.

4.3.1 Definition of local homogenization

Given $K \in \mathbb{R}^{N\times N}$, K symmetric, $A \in \mathbb{R}^{N\times M}$, $B \in \mathbb{S}^N_{++}$, $f \in \mathbb{R}^N - \{0\}$, and $\epsilon > 0$, the local homogenization problem is to find a symmetric $\widehat{K}^\dagger$ that minimizes $\mathcal{E}$, where

$$\mathcal{E}(\widehat{K}^\dagger) := \frac{1}{2}\left|\left|(K^\dagger - A\widehat{K}^\dagger A^T)f\right|\right|^2_B \; + \; \frac{\epsilon}{2}\left|\left|K^\dagger - A\widehat{K}^\dagger A^T\right|\right|^2_{F,B} ||f||^2_2 \,. \qquad (4.4)$$

The first term in the sum is $\frac{1}{2}\,||e||^2_B = \frac{1}{2}e^T Be$ and measures the error in the solution. Symbol "F, B" in second term stands for Frobenius matrix norm weighted with the matrix B. For $X \in \mathbb{R}^{N\times N}$,

$$||X||^2_{F,B} := \text{trace}(X^T BX).$$

'trace' of a matrix is the sum of its diagonal elements. If B is the identity matrix, then $||\cdot||_{F,B}$ is the standard Frobenius matrix norm.

In the limit $\epsilon \to 0$, this minimization problem can be interpreted as a method to define a local effective stiffness $\widehat{K}$ that provides the best solution

for arbitrary loads but subject to the constraint that for a particular load f the error is the smallest possible.

We choose ϵ, a dimensionless parameter, a couple of magnitudes smaller than 1. The second term is a regularization term and is included to obtain a unique $K^{\dagger}$. We require regularization for two reasons.

- Firstly, the overall solution method is iterative due to mesh refinements and Newton iteration steps. Hence f used for homogenization in a single step represents only an approximation to an "exact" f.
- Secondly, without regularization, the minimization problem given by Equation (4.4) is ill-posed (with infinitely many solutions). This happens because the action of operators K and $\widehat{K}$ is used only for a single load f. Thus regularization is necessary even if an "exact" f is known.

The "closure assumption" in this homogenization method is that local (to an element) effective material properties are not affected by other elements [5].

4.3.2 Solution of the local homogenization problem

Equation (4.4) is a typical minimization problem with linear constraints and a convex quadratic objective function in the entries of $\widehat{K}^{\dagger}$. Note that the objective function is *not* quadratic in the entries of $\widehat{K}$. In fact, because of the perturbation properties of the Moore-Penrose pseudoinverse, it is a highly nonlinear and discontinuous function in the entries of $\widehat{K}$ [24]. We will solve

for $\widehat{K}^\dagger$ and obtain $\widehat{K}$ by computing its pseudoinverse. We will see later in this section that $\widehat{K}^\dagger$ solves a continuous-time Lyapunov equation.

Symmetry of $\widehat{K}$ implies (and is implied by) symmetry of $\widehat{K}^\dagger$. We use a Lagrange multiplier matrix Λ to impose the symmetry condition. To keep the notation simple we let the unknown $\widehat{K}^\dagger$ be called X in the rest of this section. The Lagrangian is

$$\mathcal{L}(X, \Lambda) = \mathcal{E}(X) + \text{trace}(\Lambda^T(X - X^T)). \tag{4.5}$$

Stationarity of $\mathcal{L}(X, \Lambda)$ implies $X = X^T$ and

$$(A^T B A)X(A^T(ff^T + \epsilon\,||f||_2^2\, I)A) - A^T B K^\dagger(ff^T + \epsilon\,||f||_2^2\, I)A = \Lambda^T - \Lambda. \tag{4.6}$$

I is the identity matrix of size compatible with f. The details of the derivation of this equation are in Appendix A.

We define some intermediate matrices for simplification.

$$U := A^T B A \tag{4.7}$$

$$V := A^T(ff^T + \epsilon\,||f||_2^2\, I)A \tag{4.8}$$

$$W := A^T B K^\dagger(ff^T + \epsilon\,||f||_2^2\, I)A \tag{4.9}$$

Matrices U and V are symmetric. Using the definitions of U, V, and W, we can write Equation (4.6) as

$$UXV - W = \Lambda^T - \Lambda.$$

Since $\Lambda^T - \Lambda$ is skew-symmetric, so is $UXV - W$. Hence, we can eliminate Λ.

$$UXV - W + (UXV - W)^T = 0. \tag{4.10}$$

Assume $\epsilon > 0$. We then use the Sherman-Morrison-Woodbury formula [77] to express V^{-1}.

$$V^{-1} = \frac{1}{\epsilon \left|\left|f\right|\right|_2^2} \left((A^T A)^{-1} - \frac{(A^T A)^{-1} A^T f f^T A (A^T A)^{-1}}{\epsilon \left|\left|f\right|\right|_2^2 + f^T A (A^T A)^{-1} A^T f} \right) \tag{4.11}$$

This exists if the following three conditions are true. Term $\epsilon \left|\left|f\right|\right|_2^2$ is positive, $(A^T A)^{-1}$ exists, and the denominator of the second term is non-zero. The second condition holds because columns of A are linearly independent. The first and third conditions are true because $\epsilon \left|\left|f\right|\right|_2^2 > 0$ and $A(A^T A)^{-1} A^T$ is positive semi-definite. In addition, V is positive-definite since it is invertible and is sum of a positive definite matrix and a rank-1 positive semi-definite matrix.

Using $X = X^T$, invertibility of V, and symmetry of U and V, Equation (4.10) can be simplified to

$$(V^{-1}U)X + X(V^{-1}U)^T = V^{-1}(W + W^T)V^{-1}.$$

We define two more intermediate matrices.

$$C := V^{-1}U \tag{4.12}$$

$$D := V^{-1}(W + W^T)V^{-1} \tag{4.13}$$

Hence, X solves

$$CX + XC^T = D. \tag{4.14}$$

This is a continuous-time Lyapunov equation, a linear matrix equation with X as the unknown matrix. D is symmetric. If the sum of any two eigenvalues

of C is non-zero (which is satisfied if C is positive definite), then a unique solution X exists and it is symmetric [80, 4].

4.4 Locally best effective properties for all loads

We now define and solve the local homogenization problem when the load f is not given, or is 0, or we want to compute effective properties independent of any self-equilibrated load.

Given $K \in \mathbb{R}^{N\times N}$, K symmetric, $A \in \mathbb{R}^{N\times M}$, and $B \in \mathbb{S}^N_{++}$, the local homogenization problem is to find a symmetric $\widehat{K}^\dagger$ that minimizes $\mathcal{E}$, where

$$\mathcal{E}(\widehat{K}^\dagger) := \frac{1}{2}\left|\left|K^\dagger - A\widehat{K}^\dagger A^T\right|\right|^2_{F,B}. \tag{4.15}$$

The notation is same as used in Equation (4.4). We can proceed to form a Lagrangian and minimize $\mathcal{E}(\widehat{K}^\dagger)$ for a symmetric $\widehat{K}^\dagger$ as explained in Appendix A. The only difference is that compared to the definitions of the intermediate matrices in Section 4.3.2, the definitions of V and W are modified. U remains the same.

$$U := A^T B A, \tag{4.16}$$

$$V := A^T A, \tag{4.17}$$

$$W := A^T B K^\dagger A. \tag{4.18}$$

Matrices C and D are defined as they were defined in Equations (4.12)–(4.13) but in terms of these U, V, and W. We still have to solve a Lyapunov equation (Equation (4.14)).

4.4.1 A special case: minimum error in ℓ^2 norm

If we want minimum error in the ℓ^2 norm and thus choose $B = I$, it turns out that we don't have to solve a Lyapunov equation because C becomes identity. Note that in this case, $U = V$, and $W = W^T$. Thus $\widehat{K}^\dagger = X = V^{-1}WV^{-1}$.

$$\begin{aligned} \widehat{K} &= \left((A^TA)^{-1}A^TK^\dagger A(A^TA)^{-1}\right)^\dagger \\ &= \left(A^\dagger K^\dagger (A^\dagger)^T\right)^\dagger \end{aligned} \tag{4.19}$$

Compare this expression with the expression for $\widehat{K}$ derived for least error in energy norm in which $\widehat{K} = A^TKA$ (Equation (3.3)). These two expressions are different in general.

It is readily seen that if the fine-scale stiffness matrix K is positive semi-definite, then so is the coarse-scale stiffness matrix $\widehat{K}$. This can be shown as follows. If K is positive semi-definite, then so is $K^\dagger$. This implies that $x^TK^\dagger x \geq 0$ for any compatible vector x, in particular for $u = (A^\dagger)^T y$. Thus, $((A^\dagger)^T y)^T K^\dagger ((A^\dagger)^T y) \geq 0$. Rearranging, we get $y^T(A^\dagger K^\dagger (A^\dagger)^T)y \geq 0$. Thus, $A^\dagger K^\dagger (A^\dagger)^T$ is positive semi-definite. This implies that its pseudoinverse $\widehat{K} = (A^\dagger K^\dagger (A^\dagger)^T)^\dagger$ is positive semi-definite too.

4.5 Constrained convex optimization for local homogenization

In Section 4.3, the local problem to homogenize a single element was posed using a regularization approach. We created a single error function,

which was a weighted sum of the error (due to homogenization) for a specific non-zero load f and a norm of the difference of fine-scale compliance matrix ($K^\dagger$) and interpolated compliance matrix ($A\widehat{K}^\dagger A^T$). This required choosing a small weighing parameter $\epsilon > 0$. The other constraint, of symmetry of X, was imposed exactly using Lagrange multipliers.

In this section we formulate local homogenization as a (fully) constrained convex optimization problem that does not require the approximation due to regularization.

There are two sets of constraints that the entries of the unknown matrix $X = \widehat{K}^\dagger$ must satisfy exactly. Firstly, X must be symmetric. Secondly, for a specific non-zero load f the error due to homogenization should be the minimum possible. The constrained optimization formulation is derived using a two-step procedure. In the first step, the constraint equations are derived. In the second step, while satisfying the constraints of first step, X must minimize a norm of the difference of fine-scale compliance matrix ($K^\dagger$) and interpolated compliance matrix (AXA^T).

The constraint equation for minimizing the error for a specific load f can be derived from the first-order necessary optimality condition of the following problem.

$$\min_{X=X^T} \frac{1}{2} \left|\left|(K^\dagger - AXA^T)f\right|\right|_B^2$$

Using a Lagrange multiplier approach to impose $X = X^T$ as shown in Sec-

tion 4.3 and Appendix A we get the optimality condition.

$$A^T BAXA^T ff^T A - A^T BK^\dagger ff^T A \text{ must be skew-symmetric.}$$

Thus,

$$(A^T BAXA^T f - A^T BK^\dagger f)(A^T f)^T \text{ must be skew-symmetric.}$$

It can be shown that if the outer product of two vectors is skew-symmetric, then at least one of the two vectors is zero. The proof is in Appendix C. Since A is full-rank and f is assumed non-zero, it implies that $A^T f$ is non-zero. Thus the first vector, $A^T BAXA^T f - A^T BK^\dagger f$, is 0. Obviously this equation cannot be used to fully determine X since it fixes the action of X only on a single vector, $A^T f$.

This leads us to the second stage of the optimization procedure.

$$\min_X \frac{1}{2} \left|\left|K^\dagger - AXA^T\right|\right|_{F,B}^2 \text{ such that } A^T BAXA^T f = A^T BK^\dagger f \text{ and } X = X^T \tag{4.20}$$

We define auxiliary matrices U, V, W, and $\widehat{f}$, where $U = A^T BA, V = A^T A, W = A^T BK^\dagger$, and $\widehat{f} = A^T f$. Using Lagrange multipliers to impose the two constraints, we can deduce the following first-order optimality condition.

$$\begin{bmatrix} U \otimes V + V \otimes U & U \otimes \widehat{f} + \widehat{f} \otimes U \\ U \otimes \widehat{f}^T + \widehat{f}^T \otimes U & 0 \end{bmatrix} \begin{bmatrix} \text{vec}(X) \\ \mu \end{bmatrix} = \begin{bmatrix} \text{vec}(WA + A^T W^T) \\ 2\,Wf \end{bmatrix}$$

Here $\otimes$ is the Kronecker product operation, "vec" is the vectorization operation and μ is an unknown vector of Lagrange multipliers that constrain X to be the

best possible for the specific load f. The Lagrange multipliers for imposing the symmetry of X have been eliminated. The $(1,2)$ block is the transpose of $(2,1)$ block. Since the $(1,1)$ block $(U \otimes V + V \otimes U)$ is symmetric positive definite and the $(2,1)$ block $(U \otimes \widehat{f}^T + \widehat{f}^T \otimes U)$ has full row-rank, the system is invertible and characterizes the unique minimum X. The Kronecker products are not computed. One should transform the system to a Lyapunov equation like structure for fast solution. This is slightly more complicated than the regularization approach of Section 4.3. Hence, we present all the results in Chapter 6 using the simpler approach, but this derivation and the results show that a unique solution can be found for exact local homogenization.

4.6 Homogenization of pre-stress

In case of polymer lattices of SFIL, even if there is no external force on molecules, the lattice is not in equilibrium in general. This is because of the residual forces due to a mismatch in inter-molecular distance and the length at which the inter-molecular potential is minimum. Using the terminology from continuum elasticity, we call the residual forces pre-stress. If a single element is removed from the lattice, as shown in Figure 4.2, the pre-stress in that element is self-equilibrated. This is because if a spring leads to a force F on a particle, then it leads to a force $-F$ on the corresponding neighbor. Taking the sum and moment around any point, the total force as well as the total moment is zero. This pre-stress leads to a pre-strain.

To homogenize correctly, we need to transfer the pre-stress (or pre-

strain) information in the fine-scale lattice to coarse-scale mesh. This is done after determining the coarse-scale 'material properties' ($\widehat{K}$). Let p and u denote the fine-scale pre-stress and pre-strain, respectively. The corresponding coarse-scale quantities are $\widehat{p}$ and $\widehat{u}$. Figure 4.3 shows the transfer operation.

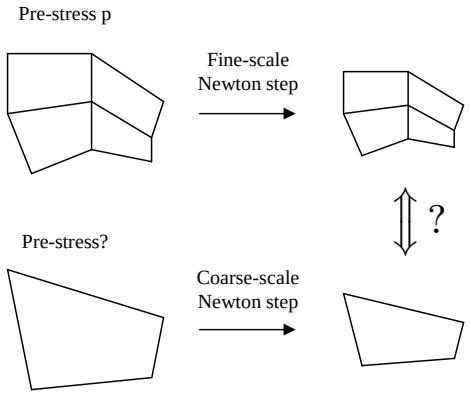


Figure 4.3: The homogenized pre-stress is the coarse-scale pre-stress that would give a deformation such that the fine-scale and coarse-scale lattices are approximately similar.

For a single iteration, we have

$$Ku = p \implies u = K^{\dagger}p$$

and

$$\widehat{K}\widehat{u} = \widehat{p} \implies \widehat{u} = \widehat{K}^{\dagger}\widehat{p}.$$

We find $\widehat{u}$ (and hence $\widehat{p}$ too) by minimizing the error

$$||A\widehat{u} - u||_B^2 ,$$

where A is the interpolation operator and B is a symmetric positive definite matrix that forms the norm in which we measure the error. We will use $B = I$

here. Since A has full column-rank, the solution is simply $\widehat{u} = A^{\dagger}u$. Using the expressions for u and $\widehat{u}$, the coarse-scale pre-stress is

$$\widehat{p} = \widehat{K}A^{\dagger}K^{\dagger}p. \tag{4.21}$$

4.7 Remarks on local homogenization

The interpolation operator A is a parameter to play with and it can be chosen to get better homogenized properties. Operator-dependent interpolation [50] should be a better choice but this is a speculation in context of the current approach. We use the simple bilinear interpolant for lattice based problems in 2-D (and trilinear in 3-D). We can also make the error $\mathcal{E}$ a function of A as well and look for the best A. This turns out to be a complicated nonlinear problem with many more unknowns.

We have a choice of the matrix norm in the regularization term. Spectral norm may be better but its use leads to a nonlinear problem with non-smooth objective function. In contrast, weighted Frobenius norm leads to a convex quadratic problem with linear equality constraints (due to the imposed symmetry of $\widehat{K}^{\dagger}$).

Load-independent homogenization (Section 4.4) may not be correct for problems involving layered media. Boundary loads and local material distribution influence the homogenized properties [2, p. 24]. For example, in conductivity problems for layered media, if the flux is in parallel to the layers, the effective conductivity is given by the arithmetic average. However, if the

flux is in the perpendicular direction, the effective conductivity is given by the harmonic average [49, p. 13]. A local self-equilibrated load can be computed by projecting the global known load.

The choice of B is critical. If we choose $B = K$ (K should be positive definite for this) then we don't gain anything when compared to the minimization in a subspace. The approximate solution of Equation (4.4) leads to the arithmetic average rather than the harmonic average. Since our interpolation operator is independent of the material properties, and we are not resolving the variations, we should look for a "weaker" convergence. Here the theory of homogenization for PDEs guides us to believe that B should be independent of the operator and correspond to a discrete approximation of the L^2 norm.

These alternatives, load dependent or independent homogenization and possible values of the matrix B, are numerically explored and justified in Section 6.2. In the next section we analytically compute effective properties in 1-D for a small element.

4.8 An analytical example of local homogenization

The local homogenization method of Section 4.3.1 is exemplified with homogenization of 3 harmonic springs forming a single element of a 1-D lattice.

We homogenize the springs to create a single effective spring with spring constant $\widehat{k}$ (Figure 4.4). To keep the analysis simple, we first choose the matrix B (used in Equation (4.4)) to be identity. Later on we use a different B to

reduce errors where we want. We recover the classical effective spring constant given by the harmonic average.

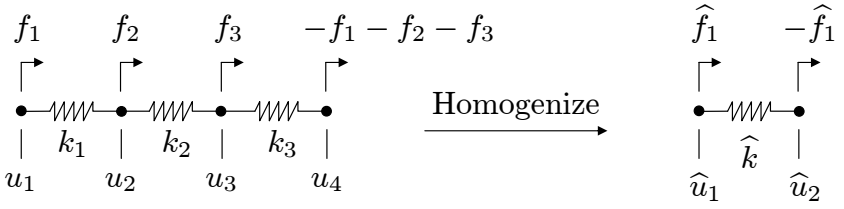


Figure 4.4: We find the best effective $\widehat{k}$ for the self-equilibrated system shown on left.

The fine-scale system is described by displacements $u \in \mathbb{R}^4$, stiffness matrix $K \in \mathbb{R}^{4\times 4}$, and load $f \in \mathbb{R}^4$ with $f_4 = -f_1 - f_2 - f_3$.

$$Ku = \begin{bmatrix} k_1 & -k_1 & 0 & 0 \\ -k_1 & k_1+k_2 & -k_2 & 0 \\ 0 & -k_2 & k_2+k_3 & -k_3 \\ 0 & 0 & -k_3 & k_3 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} = f$$

The solution is $u = K^\dagger f + (u_0, u_0, u_0, u_0)^T$ where u_0 is arbitrary and

$$K^\dagger = \frac{1}{16} \begin{bmatrix} \frac{9}{k_1}+\frac{4}{k_2}+\frac{1}{k_3} & -\frac{3}{k_1}+\frac{4}{k_2}+\frac{1}{k_3} & -\frac{3}{k_1}-\frac{4}{k_2}+\frac{1}{k_3} & -\frac{3}{k_1}-\frac{4}{k_2}-\frac{3}{k_3} \\ -\frac{3}{k_1}+\frac{4}{k_2}+\frac{1}{k_3} & \frac{1}{k_1}+\frac{4}{k_2}+\frac{1}{k_3} & \frac{1}{k_1}-\frac{4}{k_2}+\frac{1}{k_3} & \frac{1}{k_1}-\frac{4}{k_2}-\frac{3}{k_3} \\ -\frac{3}{k_1}-\frac{4}{k_2}+\frac{1}{k_3} & \frac{1}{k_1}-\frac{4}{k_2}+\frac{1}{k_3} & \frac{1}{k_1}+\frac{4}{k_2}+\frac{1}{k_3} & \frac{1}{k_1}+\frac{4}{k_2}-\frac{3}{k_3} \\ -\frac{3}{k_1}-\frac{4}{k_2}-\frac{3}{k_3} & \frac{1}{k_1}-\frac{4}{k_2}-\frac{3}{k_3} & \frac{1}{k_1}+\frac{4}{k_2}-\frac{3}{k_3} & \frac{1}{k_1}+\frac{4}{k_2}+\frac{9}{k_3} \end{bmatrix}.$$

We use standard linear interpolation for the inner points. The interpolation operator is

$$A = \begin{bmatrix} 1 & \frac{2}{3} & \frac{1}{3} & 0 \\ 0 & \frac{1}{3} & \frac{2}{3} & 1 \end{bmatrix}^T.$$

For the homogenized element with a single spring, the effective stiffness matrix is

$$\widehat{K} = \begin{bmatrix} \widehat{k} & -\widehat{k} \\ -\widehat{k} & \widehat{k} \end{bmatrix}.$$

Its pseudoinverse is

$$\widehat{K}^{\dagger} = \frac{1}{4}\begin{bmatrix} \frac{1}{\widehat{k}} & -\frac{1}{\widehat{k}} \\ -\frac{1}{\widehat{k}} & \frac{1}{\widehat{k}} \end{bmatrix}.$$

We will determine the best $\widehat{k}$ by minimizing the error $\mathcal{E}$ (Equation (4.4)). We choose $B = I$.

For an arbitrary load f and $\epsilon > 0$, we can analytically solve for the minimum. In the limit $\epsilon \to 0$ we get

$$\widehat{k} = \frac{10}{9}\frac{(3f_1 + 2f_2 + f_3)k_1k_2k_3}{3k_1k_2(f_1 + f_2 + f_3) + 3k_2k_3(f_1) + 4k_3k_1(f_1 + f_2)}.$$

To simplify the expression, let us use loads only at the end-points. Thus, $f_1 = 1, f_2 = 0$, and $f_3 = 0$. Then

$$\widehat{k} = \frac{10\,k_1k_2k_3}{9\,k_1k_2 + 12\,k_1k_3 + 9\,k_2k_3}.$$

Compare $\widehat{k}$ with the classical effective spring constant

$$k = \frac{k_1k_2k_3}{k_1k_2 + k_1k_3 + k_2k_3}.$$

The expressions for $\widehat{k}$ and k are not the same. However, we do get a value very close to the one that is given by the harmonic average. Since we chose $B = I$, the errors at the end-points were given equal emphasis as the errors at interior points. In derivation of the classical effective constant, we are only concerned with matching end-point displacements. This explains the discrepancy between the two effective spring constants.

This discrepancy can be resolved by using $B = \mathrm{diag}(M, 1, 1, M)$, where $M \gg 1$, so that errors at end-points are penalized more. The best effective

stiffness can be computed to be

$$\widehat{k} = \frac{k_1 k_2 k_3}{(1-m)k_1 k_2 + (1+2m)k_1 k_3 + (1-m)k_2 k_3}$$

where $m := \dfrac{1}{9M+1} \to 0$ as $M \to \infty$. Hence, we recover the classical effective stiffness by choosing an appropriate norm to measure the local error.

4.9 Computational aspects

Computing the homogenized material properties requires computation of Moore-Penrose pseudoinverse of multiple matrices with different structural properties. We discuss this topic briefly here in Section 4.9.1 and in detail in Chapter 5 for large sparse matrices. In addition, we have to solve a dense Lyapunov equation of a small size (number of DOFs in the coarse-scale element). This is discussed in Section 4.9.2 below.

4.9.1 Computation of Moore-Penrose pseudoinverse

The pseudoinverse of a real matrix K can be defined and computed in terms of the SVD [48, 24]. Let K have an SVD $K = U\Sigma V^T$, then the pseudoinverse is $K^\dagger = V\Sigma^\dagger U^T$ where $\Sigma^\dagger$ is formed by taking the reciprocal of all non-zero singular values (diagonal entries in Σ) and applying a transpose operation.

SVD is the most general and reliable procedure for computing pseudoinverses. However, it is an $O(N^3)$ procedure. It also ignores the sparsity of the matrix. For these reasons, it is not the best method in the context

of numerical homogenization. In Chapter 5 we analyze faster algorithms for computing the pseudoinverse. Numerical results are presented in Chapter 6. For local homogenization, the pseudoinverse is needed in the following steps.

- We need pseudoinverses of the large and sparse fine-scale stiffness matrix K. Actually, only the action of the pseudoinverse is needed on f and columns of A (Equations (4.9) and (4.18)). See Chapter 5 for details.
- In case of Equation (4.19), we need the action of $K^\dagger$ on columns of $(A^T)^\dagger$. See Chapter 5 for details.
- In case of Equation (4.19), we need the pseudoinverse of a dense A.
- Whether we homogenize for a particular load or not, we have to compute the homogenized stiffness matrix from its pseudoinverse. Thus we need to explicitly compute pseudoinverse of a symmetric matrix of a relatively small fixed size (since it is on the coarse-scale). For example, for a 3-D lattice elasticity problem with a cube forming the coarse-scale, the size of effective stiffness matrices is 24×24. In 2-D the size is 8×8.

For a rectangular matrix of full column rank, for example the interpolation matrix A, the pseudoinverse can be expressed using the normal equations.

$$A^\dagger = (A^T A)^{-1} A^T$$

Since the number of columns of A is fixed and small, we can either use the normal equations for computing $A^\dagger$ or preferably use QR factorization of A.

4.9.2 Solution of the Lyapunov equation

Computing the pseudoinverse of the best local stiffness matrix requires solving a Lyapunov equation (except for the special case mentioned above for load-independent homogenization and when B is identity). This equation arises in many different areas, for example control theory and stability analysis. Many algorithms and implementations exist for its efficient solution [70, 4]. We use the Bartels-Stewart algorithm [13] that uses lower real Schur decomposition of the matrix C. Let $C = QC'Q^T$, $D = QD'Q^T$, and $X = QX'Q^T$, where $Q \in \mathbb{R}^{M \times M}$ is a real orthonormal matrix, C' is real block lower triangular with maximum block size 2. With these definitions, Equation (4.14) becomes

$$C'X' + X'C'^{\,T} = D'.$$

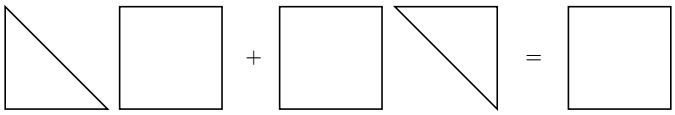


Figure 4.5: Structure of the Lyapunov equation after Schur decomposition of C.

Since C' has a special structure, the transformed equation can be solved using a "forward-substitution" algorithm. We successively solve for $X'_{11}, X'_{12}, \ldots, X'_{1m}, X'_{21}, X'_{22}$ and so on where m is the number of blocks. Finally, we can compute $X = QX'Q^T$.

4.10 Error estimation for goal-oriented adaptivity

We presented goal-oriented adaptivity for self-adjoint elliptic problems in Section 3.3. We had used Galerkin orthogonality in the derivation of local error estimates. The orthogonality condition is slightly different if the fine-scale and coarse-scale operators are different. This is the case if the coarse-scale operator is derived by local homogenization. We derive the corresponding error estimates here. As before, quantities relevant on the coarse-scale are marked with a hat and A is a global interpolation operator from the coarse-scale to the fine-scale. We assume that the coarse-scale is chosen in such a way that the Dirichlet boundary conditions are satisfied exactly.

The fine-scale primal problem is

$$\text{Find } u \in u_D + \mathcal{V} \text{ such that } \mathcal{B}(u, v) = \mathcal{L}(v) \quad \forall\, v \in \mathcal{V}.$$

The coarse-scale primal problem is

$$\text{Find } \widehat{u} \in u_D + \widehat{\mathcal{V}} \text{ such that } \widehat{\mathcal{B}}(\widehat{u}, \widehat{v}) = \widehat{\mathcal{L}}(\widehat{v}) \quad \forall\, \widehat{v} \in \widehat{\mathcal{V}}$$

where the coarse-scale load is defined by

$$\widehat{\mathcal{L}}(\widehat{v}) = \mathcal{L}(A\widehat{v}) \quad \forall\, \widehat{v} \in \widehat{\mathcal{X}}.$$

Corresponding to a goal $\mathcal{G} \in \mathcal{X}'$, we have the fine-scale and coarse-scale adjoint problems. The fine-scale adjoint problem is

$$\text{Find } w \in \mathcal{V} \text{ such that } \mathcal{B}(v, w) = \mathcal{G}(v) \quad \forall\, v \in \mathcal{V}.$$

The coarse-scale adjoint problem is

$$\text{Find } \widehat{w} \in \widehat{\mathcal{V}} \text{ such that } \widehat{\mathcal{B}}(\widehat{v}, \widehat{w}) = \widehat{\mathcal{G}}(\widehat{v}) \quad \forall\, \widehat{v} \in \widehat{\mathcal{V}}$$

where the goal on the coarse-scale is defined by

$$\widehat{\mathcal{G}}(\widehat{v}) = \mathcal{G}(A\widehat{v}) \quad \forall\, \widehat{v} \in \widehat{\mathcal{X}}.$$

The error in the goal is

$$\begin{aligned}
\mathcal{G}(u) - \widehat{\mathcal{G}}(\widehat{u}) &= \mathcal{G}(u) - \mathcal{G}(A\widehat{u}) \\
&= \mathcal{G}(u - A\widehat{u}) \\
&= \mathcal{B}(u - A\widehat{u}, w) \\
&= \mathcal{B}(u - A\widehat{u}, w - A\widehat{w}) + \mathcal{B}(u - A\widehat{u}, A\widehat{w}) \\
&= \mathcal{B}(u - A\widehat{u}, w - A\widehat{w}) + \mathcal{B}(u, A\widehat{w}) - \mathcal{B}(A\widehat{u}, A\widehat{w}).
\end{aligned}$$

In case of compatible fine-scale and coarse-scale bilinear forms, the last two terms would have canceled each other. This is not true for our case. Instead, we prove that $\mathcal{B}(u, A\widehat{w}) = \widehat{\mathcal{B}}(\widehat{u}, \widehat{w})$ and simplify the error expression.

$$\begin{array}{rlcll}
 & \mathcal{B}(u, v) & = & \mathcal{L}(v) & \forall\, v \in \mathcal{V} \\
\Rightarrow & \mathcal{B}(u, A\widehat{v}) & = & \mathcal{L}(A\widehat{v}) & \forall\, \widehat{v} \in \widehat{\mathcal{V}} \\
\Rightarrow & \mathcal{B}(u, A\widehat{v}) & = & \widehat{\mathcal{L}}(\widehat{v}) & \forall\, \widehat{v} \in \widehat{\mathcal{V}} \\
\Rightarrow & \mathcal{B}(u, A\widehat{v}) & = & \widehat{\mathcal{B}}(\widehat{u}, \widehat{v}) & \forall\, \widehat{v} \in \widehat{\mathcal{V}} \\
\Rightarrow & \mathcal{B}(u, A\widehat{w}) & = & \widehat{\mathcal{B}}(\widehat{u}, \widehat{w}) &
\end{array}$$

Finally, the error in the goal, $\mathcal{G}(u) - \widehat{\mathcal{G}}(\widehat{u})$, is

$$\underbrace{\mathcal{B}(u - A\widehat{u}, w - A\widehat{w})}_{\text{Standard characterization}} + \underbrace{\widehat{\mathcal{B}}(\widehat{u}, \widehat{w}) - \mathcal{B}(A\widehat{u}, A\widehat{w})}_{\text{Incompatible bilinear forms}}.$$

As presented in Section 3.3, we can split both the terms into quantities defined on individual elements. As before, the expression above uses the exact primal and adjoint solutions. To approximate the error estimates, we work with a current coarse mesh and a uniformly refined finer mesh as an approximation of the fine-scale mesh. These local error estimates provide indicators for a further mesh refinement.

4.11 Integration of homogenization, mesh adaptivity, and Newton iterations for nonlinear problems

We summarize the basic structure for homogenizing and solving the problem on a coarse mesh. Figure 4.6 depicts this structure in a flowchart.

We create a compatible coarse mesh from the fine lattice data. Here compatibility means that all fine lattice Dirichlet boundary conditions are reproducible by the coarse mesh Dirichlet boundary conditions. After making an initial guess for the displacements, we compute the gradient of energy on the fine lattice and restrict (using the transpose of the interpolation operator) it to the coarse lattice. We continue if the norm of the gradient is large, otherwise we stop and report convergence. The next step is to compute the Hessian on the coarse lattice by homogenizing individual coarse elements. Using the coarse gradient and Hessian we perform the Newton step for coarse displacements. The new displacements are prolongated to the fine lattice, and we iterate for the next step. Once the step size is below a threshold, we use energy-oriented or goal-oriented error estimates and refine elements with large errors. We

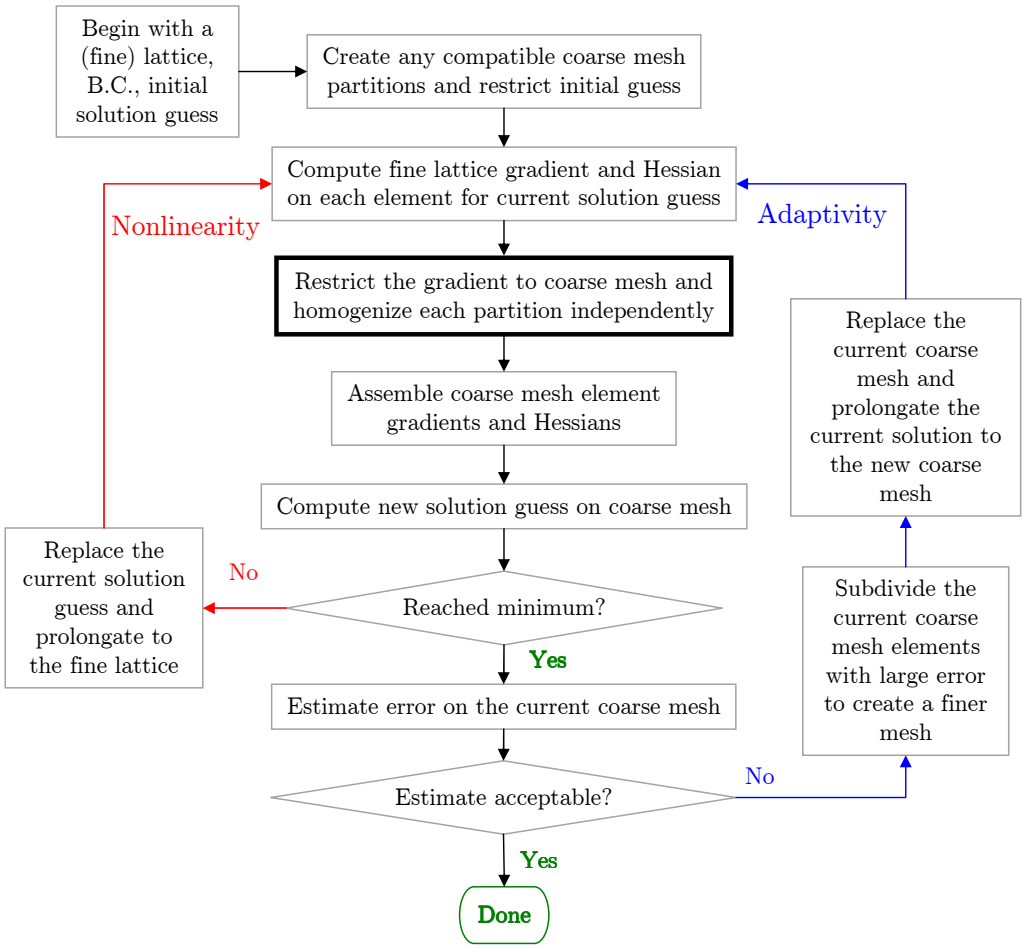


Figure 4.6: Overall structure of integrating homogenization, mesh adaptivity, and Newton iterations for nonlinearity.

repeat the Newton iterations on the refined lattice until convergence. The mesh is refined again if needed.

Chapter 5

Fast Algorithms for Moore-Penrose Pseudoinverse

Local homogenization is the core part of the global solution method and is done in each element. Moreover, because of mesh refinement and nonlinearities, each element can possibly have different stiffness matrix after each step, and it may be required to upscale it again. Thus, critical to the computational efficiency of the whole method is fast and approximate computation of the pseudoinverse of a sparse element stiffness matrix. For these reasons, Singular Value Decomposition (SVD) is not the best choice here for computing pseudoinverses.

We present algorithms that are significantly faster for sparse matrices. The first one uses the characterization of the pseudoinverse as the limit of a Tikhonov regularized matrix [79, 19]. In the second algorithm, the *a priori* knowledge of the null-space of a matrix can be used to compute the pseudoinverse by direct or iterative linear solvers [47]. The third algorithm uses a sparse rank-revealing QR factorization of the matrix (or of a suitable column permutation) to compute an *exact* pseudoinverse using the matrix factors Q and R [24, 34]. Lastly, in the context of mesh refinement or nonlinear prob-

lems, one might reuse an old pseudoinverse (actually its sparse factorized form) to compute pseudoinverse of a perturbed matrix. This can be done using an iterative procedure based on "proper splittings" [22, 66].

None of these algorithms is the perfect one for all situations. Depending on the problem dimension, number of vector unknowns per DOF, element size, condition numbers, and sparsity, one algorithm may be much better than the others. These choices are analyzed in Sections 6.5 and 6.6.

5.1 Moore-Penrose pseudoinverse

Moore-Penrose pseudoinverse was discovered by E. H. Moore in 1906 in the context of projections associated with singular and rectangular matrices [63]. It was rediscovered by Penrose in 1955 as the unique matrix satisfying four algebraic matrix equations [73]. Quite a few articles and books have appeared on the subject since then. There is a large bibliography in [67] and [19].

Moore-Penrose pseudoinverse is the most widely known special case of *generalized inverses* or pseudoinverses. In this dissertation we exclusively work with Moore-Penrose pseudoinverse and occasionally use the term pseudoinverse for it.

5.2 Computation of Moore-Penrose pseudoinverse

Since pseudoinverses are generalizations of inverses of square nonsingular matrices, their computation usually involves solving linear equations or

obtaining matrix factorizations. Analogous to the plethora of direct and iterative algorithms for solution of linear systems of equations, there is no single algorithm that is the best choice for computing generalized inverses. There is a variety of algorithms that can be used [19]. We focus only on a few that are most likely to be useful for our purposes.

One of the most general and reliable computational method to compute Moore-Penrose pseudoinverse is the SVD [48, 24]. The pseudoinverse of a real matrix K can be defined and computed in terms of the SVD [48, 24]. Let K have an SVD $K = U\Sigma V^T$, then the pseudoinverse is $K^\dagger = V\Sigma^\dagger U^T$ where $\Sigma^\dagger$ is formed by taking the reciprocal of all non-zero singular values (diagonal entries in Σ) and applying a transpose operation. However, SVD is an $O(N^3)$ procedure. Typical implementations also ignore the sparsity of the matrix. For these reasons, SVD is not the best method in the context of homogenization. However, SVD is useful for determination of the numerical rank, which is important to reliably compute pseudoinverse of arbitrary matrices.

Despite its excellent numerical properties, use of SVD gives an impression that computation of pseudoinverse is *inherently* an iterative process just like the computation of singular values or eigenvalues. This impression is not correct. Entries of the pseudoinverse matrix are explicit rational functions of the entries of the original matrix just like the entries of the inverse of an invertible matrix. This is readily seen from the equivalent definition of the pseudoinverse that uses Tikhonov regularization [19] or full-rank factorizations. Looking just at Tikhonov regularization, the entries of the inverse of

a non-singular matrix are rational polynomials in the entries of the original matrix. Thus, the entries of the pseudoinverse matrix are limits of rational polynomials expressions. In principle, they can be expressed as explicit functions and are exactly computable unlike eigenvalues or singular values. We do not use this observation for computing pseudoinverses, but mention it because of its conspicuous absence in the literature. Moreover, this gives a hint that if we know the rank of the matrix *a priori*, possibly using physical considerations, then better algorithms can be devised, for example the one in [47].

5.3 Pseudoinverses and homogenization

Our focus is on computing Moore-Penrose pseudoinverses of the following three categories of matrices.

1. A real, singular (hence square), sparse, symmetric matrix with almost all eigenvalues being positive. The rank-deficiency is known in advance and is small (typically between 1 and 6) and is independent of the matrix size. The matrix size is variable, typical value being between 500 and 3000. These are element stiffness matrices coming from a finite element discretization.

2. A real, dense, rectangular, full column-rank matrix with a small number of columns, between 2 and 24, and a variable number of rows, typical value being between 500 and 3000. These are interpolation matrices in a finite element discretization.

3. A real, singular (hence square), dense, symmetric matrix. The matrix size is small and will be between 2 and 24. These are pseudoinverses of homogenized fine-scale matrices.

For the second category (the dense, rectangular, full column-rank matrices) the pseudoinverse can be expressed and computed using the normal equations.

$$A^{\dagger} = (A^T A)^{-1} A^T$$

Since the number of columns of A is fixed and small and A is well-conditioned, we can either use the normal equations for computing $A^{\dagger}$ or preferably use QR factorization of A to avoid loss of precision due to squaring of the condition number. For the third category (the small, dense, symmetric, singular matrices), we use the SVD. These are implemented in LAPACK [3].

The computational bottleneck of the homogenization process is computing the pseudoinverses of the matrices of the first category (real, large, sparse, singular matrices). SVD of a matrix of size 400 takes nearly a second on current single core processor (as of year 2009). However, a sparse Cholesky factorization of matrices of similar size and with a structure and properties of the homogenization problem takes around 0.005 seconds, which is 200 times faster. If the matrix size is greater, say 2500, the sparse factorization can be 3000 times faster than computing the SVD. Thus, if we can use sparse factorization or sparse solution methods for linear systems instead of SVD for pseudoinverse, we can hope to obtain a speedup of 100 to 1000 depending on the matrix size.

Note that when we talk about computing pseudoinverse, in general we do not want it in an explicit matrix form but want only its action on a set of vectors. However, for the categories of matrices mentioned above, we *do* want the pseudoinverse of the dense matrices explicitly. This is all right, since these are small matrices.

5.4 Sparse algorithms for Moore-Penrose pseudoinverse

As mentioned above, using a sparse algorithm for computing pseudoinverse may be 2 or 3 orders of magnitude faster than the SVD for typical sparse finite element matrices. We present three algorithms that exploit the sparsity. In the end, we present an iterative algorithm that can compute pseudoinverse of a perturbed matrix using the factorized form of the pseudoinverse of the original matrix. This is useful when the element stiffness matrices change slightly during mesh adaptivity steps and Newton step for nonlinearity.

5.4.1 Pseudoinverse using Tikhonov regularization

We can avoid the SVD by using a limit characterization for the pseudoinverse using Tikhonov regularization [79, 19]. For an arbitrary matrix K

$$K^{\dagger} = \lim_{\delta \to 0} (K^T K + \delta I)^{-1} K^T = \lim_{\delta \to 0} K^T (K K^T + \delta I)^{-1}. \tag{5.1}$$

I is a size-compatible identity matrix. The limit always exists. The limit has to be taken for the *full* expression and not just the matrix with the variable δ ($(K K^T + \delta I)^{-1}$ or $(K^T K + \delta I)$). The proof is in [79, 19].

5.4.1.1 Approximation using a finite δ

We use a finite δ in Equation (5.1) to approximate $K^\dagger$. For an accurate approximation, δ should be a small multiple of the σ_r^2, the square of the smallest non-zero singular value of K. We can then use Equation (5.1) to compute an approximate action of $K^\dagger$ on any given vector g. An *a priori* error estimate for $\delta > 0$ can be proved [25, 19]

$$\left|\left|(K^T K + \delta I)^{-1} K^T - K^\dagger\right|\right|_2 \leq \delta \left|\left|K^\dagger\right|\right|_2^3 .$$

We use the sparse direct Cholesky factorization method as implemented in CHOLMOD [33]. If K is sparse or banded then KK^T and $K^T K$ are sparse too (although less so). For example, if K is tridiagonal, then KK^T and $K^T K$ are pentadiagonal. Moreover, they are positive semi-definite. Addition of a positive diagonal makes the matrix to be inverted positive definite. Thus, Cholesky factorization is a feasible algorithm for all K.

It is not necessary to use a general sparse factorization method for the matrices we are interested in. If a typical nodal ordering is used to order particles in a 2-D or 3-D box, the stiffness matrices are banded. For n^d particles and p DOFs per particle, where d is the space dimension and n is the "edge-size" of an element, the matrix size is $p\,n^d$ and bandwidth is $O(p\,n^{d-1})$. Thus banded Cholesky solvers can also be competitive with general sparse Cholesky solvers. Currently, we have used reordering based sparse direct solvers. A brief comparison of the two methods is done in Section 6.5.

Because δ is finite, we lose a few digits in the computed vector $K^{\dagger}g$. If δ is chosen appropriately, then the approximation is not detrimental to the overall algorithm. It should not be too small (relative to σ_r^2) either since it would lead to numerical singularity of $K^TK + \delta I$. A large δ would lead to less accuracy. We *do* need an estimate of σ_r^2 for this problem, which is an $O(N^3)$ procedure for an accurate estimate, but that is all right since it can be done once and re-used for other elements.

This algorithm squares the (pseudo-)condition number by computing K^TK (or KK^T), and then adds a small diagonal perturbation to make the matrix invertible. Hence, we do not use an iterative algorithm to invert. One can use the QR factorization to compute the pseudoinverse for this *damped least squares problem* [24] and avoid squaring of the condition number, but it would be slower than sparse Cholesky factorization. And if one goes through the trouble of implementing sparse QR factorization, one might as well compute the pseudoinverse "exactly" using the algorithm presented in Section 5.4.3. The advantage of sparse Cholesky factorization is its speed. For certain solution regimes, this is the fastest of all the algorithms we will discuss. It is inherently an approximate algorithm and will never obtain solutions with full accuracy. This is not a big drawback because in the context of homogenization with mesh adaptivity and nonlinearity, we need only an approximation of the upscaled Hessian matrix and not its exact upscaling.

5.4.1.2 Iterative improvement using a series representation

The pseudoinverse can also be represented by the sum of a series [18].

$$K^{\dagger} = \sum_{i=1}^{\infty} K^T (KK^T + I)^{-i} \tag{5.2}$$

Here K^T must not be removed as a factor from the series (else the sum will diverge). Using the property $(\alpha K)^{\dagger} = \frac{1}{\alpha} K^{\dagger}$ for $\alpha \neq 0$, it can be derived from Equation (5.2) that

$$K^{\dagger} = \sum_{i=1}^{\infty} K^T \delta^{i-1} (KK^T + \delta I)^{-i} \ \ \forall \delta > 0.$$

Here $\delta = \frac{1}{\alpha^2}$. Since $\alpha \neq 0$ is arbitrary, the representation is valid for all $\delta > 0$. Note that the first term in the summation is the last expression in Equation (5.1) for a finite δ.

This representation effectively rescales the matrix K so that computing the inverse of $KK^T + \delta I$ is not too badly-conditioned when K is rank-deficient (and δ is chosen appropriately). A large δ would mean that the inverse can be accurately computed, but then more terms in the series will be needed to accurately approximate $K^{\dagger}$. If a smaller δ is chosen, then fewer terms will be needed to achieve an acceptable accuracy, but the inverse will not be accurately computed in finite precision. Thus, there is a built-in trade-off between computation speed and high accuracy.

Once $KK^T + \delta I$ is factorized using a sparse Cholesky factorization, the factors can be reused to compute the actions of $(KK^T + \delta I)^{-i}$ for $i \geq 1$. Thus,

the "inversion" step is required only once and rest of the terms can computed cheaply.

We have not used this iterative improvement strategy in the context of homogenization, but it can be used when the pseudoinverse is required with higher accuracy.

5.4.2 Pseudoinverse using a known null-space basis

A less well-known result, presented in [47], is that computing pseudoinverse, or its action on a vector, can be transformed to solving linear algebraic equations of same size (by either direct or iterative algorithms). One needs the null space of the original matrix *a priori*. For many mathematical models of physical problems the null-space is known from physical arguments.

For nonlinear lattice elasticity problems, the null-space of the stiffness matrix contains the rigid body translations whether the lattice is in equilibrium or not. If it is in equilibrium, rigid body rotations are also in the null-space [47]. Thus, the dimension of the null-space is independent of the size of the lattice, and its basis is known.

Let $R \in \mathbb{R}^{N\times P}, 0 < P < N$, be a matrix of columns-vectors that form an orthonormal basis of the null-space of a symmetric $K \in \mathbb{R}^{N\times N}$. R stands for rigid body motion. Thus, $P := I_n - RR^T$ is the orthogonal projector on the range of K. It is shown in [47] that $P(K + RR^T)^{-1} = (K + RR^T)^{-1}P$ is the Moore-Penrose pseudoinverse of K.

Hence, if K and R are known, then computing $u = K^{\dagger}g$, for $f \in \mathbb{R}^N$,

implies solving $(K+RR^T)u = Pg$ for u. Even if K is sparse, $(K+RR^T)$ would be dense in general. But it is a rank p update of a sparse matrix and p is small compared to N. Thus the action of $(K+RR^T)$ on any vector can be computed with a little more effort than sparse matrix vector multiplication. Thus the system can be efficiently solved by iterative solvers for symmetric matrices. We use the iterative solvers (conjugate gradient and minimum residual methods) and preconditioners (Jacobi and Incomplete Cholesky) implemented in PETSc [11]. A numerical comparison of these 4 combinations is presented in Chapter 6.

5.4.3 Pseudoinverse using QR factorization

Let $K \in \mathbb{R}^{N\times N}$ have a rank-deficiency p, where $0 < p < n$. In our application, p will be a fixed number much smaller than n. We can compute a rank-revealing QR factorization $K = QR$, where Q is orthonormal and R is upper triangular. Partition Q and R as follows.

$$Q = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix}.$$

and

$$R = \begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \end{bmatrix}.$$

Here Q_1 is $n\times(n-p)$, Q_2 is $n\times p$, R_{11} is $(n-p)\times(n-p)$, and R_{12} is $(n-p)\times p$.

Define auxiliary matrices S and X.

$$S := R_{11}^{-1}R_{12}$$

and

$$X := (S^T S + I)^{-1} S^T R_{11}^{-1} Q_1^T.$$

S is $(n-p) \times p$, and X is $p \times n$. The action of R_{11}^{-1} can be computed by back-substitutions since R_{11} is a triangular matrix. In addition, the 'inversion' of a $p \times p$ system $(S^T S + I)$ is needed to compute S and X.

Finally, using the QR factorization and the matrices S and X, it can be shown [24]

$$K^\dagger = \begin{bmatrix} R_{11}^{-1}(Q_1^T - R_{12}X) \\ X \end{bmatrix}.$$

Note that this is just an expression and need not be computed explicitly. It can be used to compute $K^\dagger f$ for a known vector f. One should form $Q_1^T f$ first to compute X and $K^\dagger f$.

In general the factors Q and R are not as sparse as the original matrix K. Moreover, a typical QR factorization algorithm will not exploit the sparsity of K. For this, we can use a sparse QR factorization algorithm, for example as implemented in [34].

The columns of K should be reordered before the QR factorization to reduce fill-in. Thus, we need to form the Q and R factors of KP, where P is a permutation matrix, usually chosen automatically [34]. If $KP = QR$, where P is orthonormal, then it can be shown using SVD that $K^\dagger = P(QR)^\dagger$. This expression is applicable for a permutation matrix P also because permutations are orthonormal. $(QR)^\dagger$ can be computed as shown above. The expressions remain valid but the implementation takes advantage of the sparsity of Q and

R.

5.4.4 Pseudoinverse using proper splittings

In the context of homogenization for mesh-adaptivity and nonlinearities, each of the algorithms to compute Moore-Penrose pseudoinverse presented above recomputes the pseudoinverse of the element stiffness matrix whenever needed. However, in some cases, material nonlinearity might not be strong enough to change the element stiffness matrix appreciably when the mesh is refined away from the element of interest. Hence, once pseudoinverse is computed in an earlier step, its value may be useful in the new step if we do not care for an exact upscaling. In a few steps, one could update the pseudoinverse to avoid slower convergence in Newton steps (if observed). With this motivation, we can use an algorithm presented in [22, 66] to update the pseudoinverse of a perturbed matrix by reusing the factorized form of the old pseudoinverse.

Analogous to additive splittings for solving nonsingular system of equations, one can use a *proper splitting* for computing pseudoinverses. A splitting $K = G - H$ of a matrix K is called proper if the range and null spaces of K and G are equal. Using such a splitting, the iterative algorithm

$$u^{(n+1)} = G^{\dagger} H u^{(n)} + G^{\dagger} f$$

converges to $u = K^{\dagger} f$ as $n \to \infty$ iff the spectral norm of $G^{\dagger} H$ is less than one.

This iteration can be useful if we know $G^{\dagger}$, the old pseudoinverse, in factorized form using Cholesky factorization or QR factorization discussed

above. H is (negative of) the update of the element stiffness matrix due to a new solution guess. One must make sure that the range and null space of K and G are identical. In many cases this can be readily done using physical arguments. In addition, if the update is small, only a few iterations will be needed to converge to u.

Despite its appeal, this algorithm requires an independent algorithm that computes the factorization of the pseudoinverse. Moreover, it is memory-intensive because of extra storage that will be useful for just a few steps. Finally, to ensure convergence *a priori* one needs an estimate of the spectral norm of $G^{\dagger}H$ or abort the computation if norm of $u^{(n)}$ is diverging to infinity. We have not pursued this approach further in this dissertation. We mention it because using this method may lead to faster computation at the expense of more memory usage.

Chapter 6

Numerical Results

This chapter contains the numerical results of integration of the local numerical homogenization method with mesh adaptivity for nonlinear lattice elasticity problems. The results provide evidence of the accuracy, robustness in presence of nonlinearities and mesh adaptivity, and fast computational speed of the method. As mentioned in Section 4.10, this method extends the existing goal-oriented h-refinement strategy [39] to numerical homogenization where coarse-scale and fine-scale operators are different. The adjoint solutions on coarse and fine meshes provide a basis of automatic goal-oriented adaptivity. This gives rise to 1-irregular meshes with hanging nodes. These are handled using the constrained approximation techniques. We present the details for 2-D and 3-D SFIL geometries in Section 6.8.

Before presenting the results on the full lattice geometries, we present numerical results of the intermediate steps in detail. The homogenization method is verified using a 2-D chessboard conductivity problem with a known homogenized limit [56]. We compute the optimum element size for homogenization of SFIL lattice. Optimum size means one that leads to fastest global homogenization without taking loss of accuracy (due to large elements) into

consideration. Some of the algorithms for Moore-Penrose pseudoinverse discussed in Chapter 5 are analyzed for computational time as a function of space dimension, element size, iterative method, and preconditioner. We compare the computational times of the sparse Cholesky factorization method for approximate pseudoinverses with the exact QR factorization method. Based on these results, we have created a feature matrix for different pseudoinverse methods that could be useful for other applications.

6.1 Verification of numerical homogenization

We verify the local numerical homogenization algorithm presented in Chapter 4 using a 2-D chessboard conductivity problem with a known homogenized limit [56]. If the conductivities of individual boxes of different colors are k_1 and k_2, then the effective conductivity is $\sqrt{k_1 k_2}$ as box-width $\epsilon \to 0$, Figure 6.1. This conductivity pattern has been used for operator-dependent multigrid homogenization [58, 65]. However, in both the articles, the error due to homogenization was large when the two conductivities differed significantly.

Using this analytical result, our goal is to form a pure Neumann problem with finite ϵ that also has a *computable* definition of effective conductivity. We discretize this problem using finite elements and solve the resulting system exactly. This gives us a value of effective conductivity that will, in general, be different than $\sqrt{k_1 k_2}$ because of finite ϵ and discretization error. We then apply the local numerical homogenization method based on the Moore-Penrose pseudoinverse of the stiffness matrix and compute an effective conductivity. A

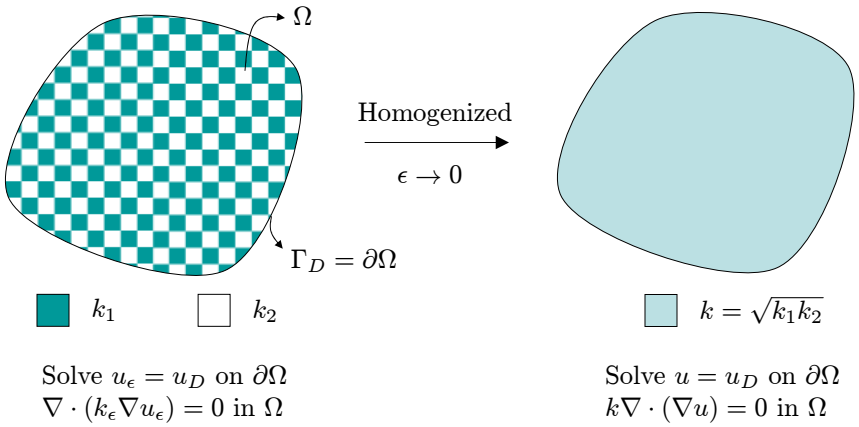


Figure 6.1: A 2-D domain with chess-board pattern of conductivities k_1 and k_2 has a limiting conductivity $\sqrt{k_1 k_2}$ as the width of the boxes $\epsilon \to 0$.

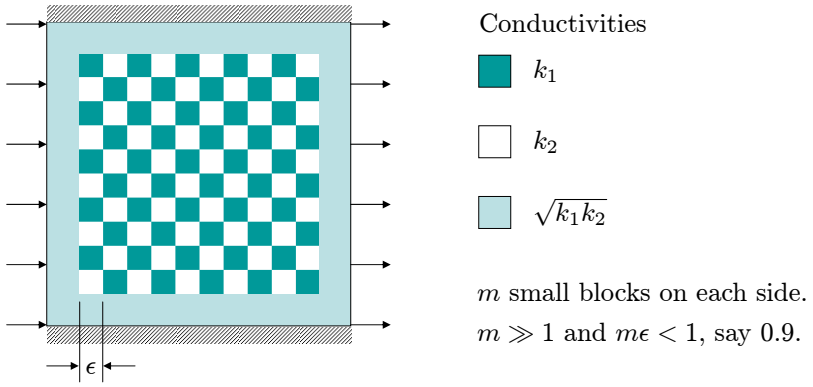


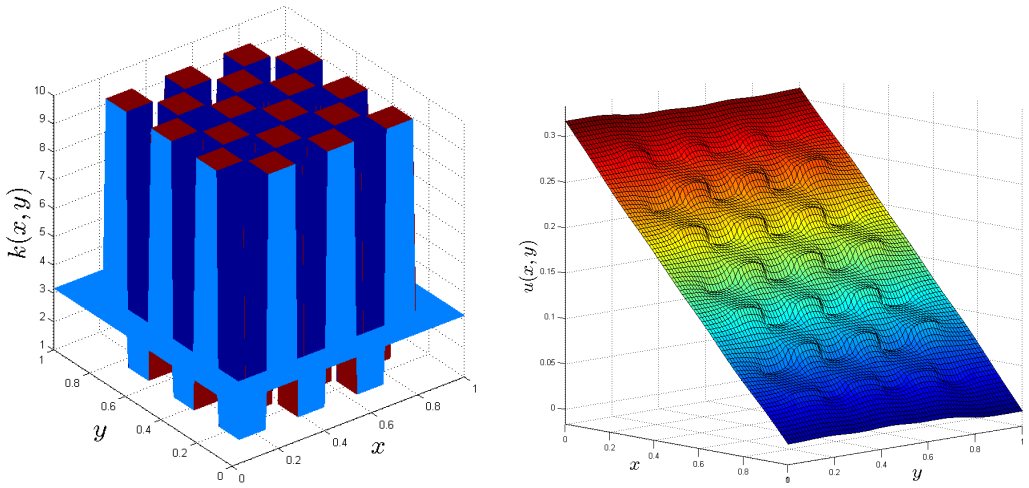
Figure 6.2: A square box of width 1 and flux boundary conditions on all edges.

comparison of these two effective conductivities will show whether the homogenization method is acceptable or not.

The pure Neumann problem is posed on a square domain of width 1 and a chess-board conductivity pattern with conductivities k_1 and k_2 in the interior, Figure 6.2. Uniform heat flows in from the left edge and flows out

from the right edge. The magnitude of the flux is f. The top and bottom edges are insulated. To avoid boundary effects a thin layer of material near the boundary has conductivity equal to the homogenized limit $\sqrt{k_1 k_2}$. Since this is a pure Neumann problem, we have to fix a datum. We choose the minimum temperature in the domain to be exactly 0.

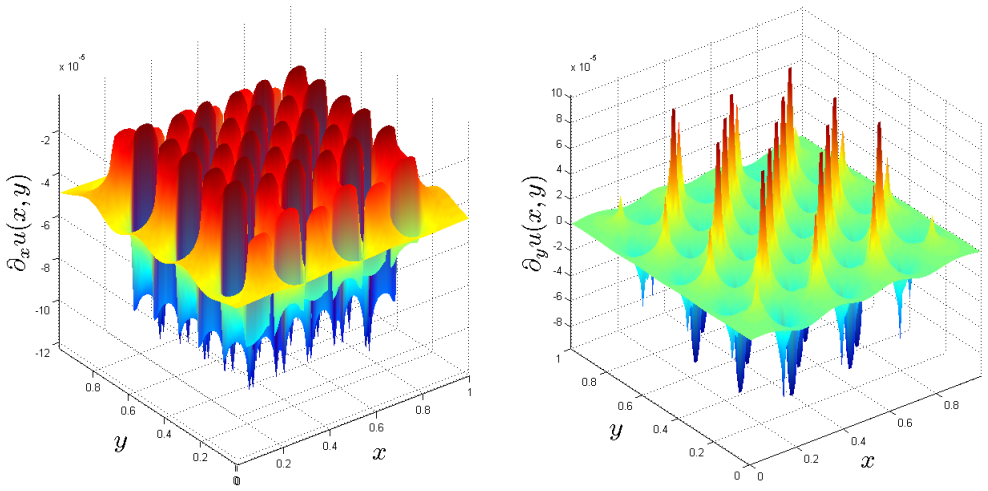
Figure 6.3(a) and Figure 6.3(b) show the conductivity pattern as a function of x and y and the temperature profile for 8 small boxes on each side ($\epsilon = 0.125$) and each small box discretized by 10 bilinear quads. Conductivity values are $k_1 = 1$ and $k_2 = 10$ and magnitude of imposed flux is 1.



(a) **(b)**

Figure 6.3: (a) The conductivity pattern for $k_1 = 1$ and $k_2 = 10$ and $\epsilon = 0.125$ in a box of width 1. (b) The plot shows a finite element solution for the temperature when the magnitude of the imposed flux is 1 and the square is discretized by 80×80 bilinear quad elements.

The finite element solution shown in Figure 6.3(b) looks smooth. However, the analytical solution has singularities at all interior corners where the boxes of different conductivities meet. This is made clear by the finite element x and y derivatives of the temperature in Figures 6.4(a) and (b) respectively.



(a) **(b)**

Figure 6.4: (a) The finite element x derivative of the function shown in Figure 6.3(b). (b) The finite element y derivative of the same function.

From the analytical homogenization result (as $\epsilon \to 0$) it is seen that the conductivity of the whole domain will be effectively $\sqrt{k_1 k_2}$. The limiting 2-D problem can be solved analytically as a 1-D problem. There will be no temperature variation in the vertical direction. The limiting solution will vary linearly in the horizontal direction.

The exact solution in the limit is

$$u(x,y) = (1-x)\frac{f}{\sqrt{k_1 k_2}}, \tag{6.1}$$

where u denotes the temperature, and f is the magnitude of boundary flux. Using Equation (6.1), the maximum temperature in the box is at $x = 0$ and is equal to $\frac{f}{\sqrt{k_1 k_2}}$. Thus, a computable definition of effective conductivity is

$$k_{\text{eff}} := \frac{f}{\max_{x,y}(u(x,y)) - \min_{x,y}(u(x,y))} = \frac{f}{\max_{x,y}(u(x,y))}. \tag{6.2}$$

We compute k_{eff} for the finite element solution also. The local numerical homogenization method is used to upscale the whole domain into a single quad with bilinear shape functions. The output is a coarse-scale symmetric "stiffness" matrix of size 4 (the number of DOFs in the coarse-scale element). We can then extract the best conductivity from this stiffness matrix and compare it with the k_{eff} for the finite element solution using Equation (6.2). Denote the numerically homogenized conductivity k_ϵ (because it will vary with ϵ).

For a bilinear quad with conductivity k and vertex nodes ordered as [1 = (0,0), 2 = (1,0), 3 = (0,1), 4 = (1,1)], the element stiffness matrix is

$$K = kK_0 \text{ where } K_0 = \frac{1}{6}\begin{bmatrix} 4 & -1 & -1 & -2 \\ -1 & 4 & -2 & -1 \\ -1 & -2 & 4 & -1 \\ -2 & -1 & -1 & 4 \end{bmatrix}$$

The numerical homogenization algorithm will give the coarse-scale matrix $\widehat{K}$ of size 4×4. In general it will not be proportional to K so we cannot extract

a conductivity k_ϵ for making a comparison. We use Frobenius matrix norm to get the best k_ϵ such that $\left|\left|K - \widehat{K}\right|\right|_F^2$ is minimum. This gives

$$k_\epsilon = \frac{K_0 : \widehat{K}}{K_0 : K_0} = \frac{\text{trace}(K_0^T \widehat{K})}{\text{trace}(K_0^T K_0)}$$

where ':' is the matrix inner product induced by the Frobenius norm and 'trace' is the matrix trace. Note that this extraction of a single number k is just for the comparison purposes of this section. For finite element assembly in a "real" problem we use the matrix $\widehat{K}$ obtained after homogenization.

For a fixed $k_1 = 1$ and variable $k_2 > 1$, Figure 6.5 shows the comparison between the limiting conductivity ($\sqrt{k_2}$) and k_ϵ produced by the homogenization method. k_2 is varied from 1 to 100. The curves show that for this range and a fixed number of elements (1600) to resolve the solution, the two values are quite close. The difference increases for large contrast between k_1 and k_2 but it may be possible to rectify that by using smaller and more elements in the mesh to resolve the singularities in the solution derivatives. We do not attempt that here.

6.2 Alternatives in homogenization error functional

In Section 4.3 we formulated the local numerical homogenization problem when a local self-equilibrated load is known. In Section 4.4 we formulated the same problem by treating all loads equally. In both cases we had used a norm (given by a matrix B) to compare the local fine-scale and coarse-scale compliance matrices. Thus, we have to make two choices − whether a known

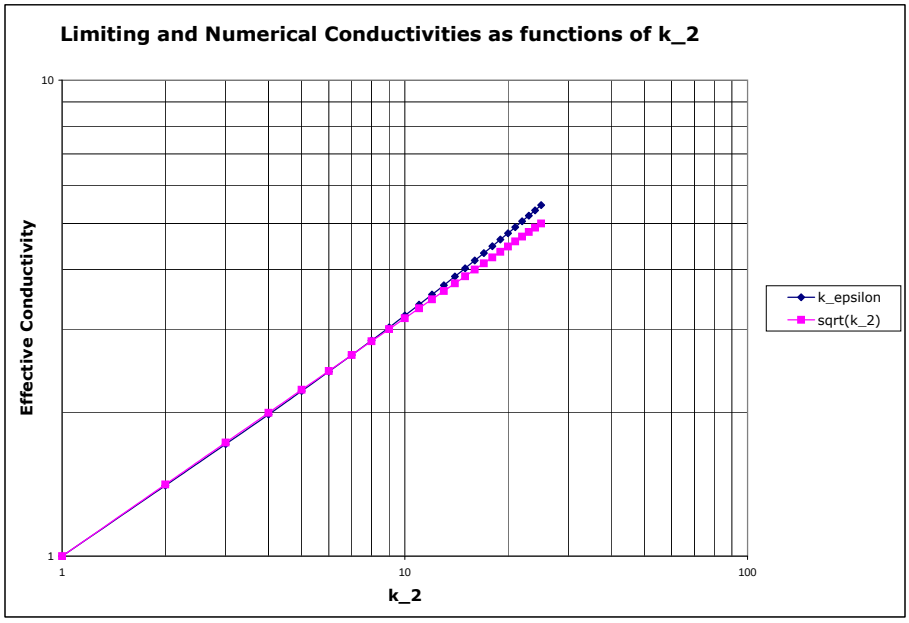


Figure 6.5: A comparison between limiting conductivity $\sqrt{k_1 k_2}$ and homogenized conductivity k_ϵ as k_2 is varied and $k_1 = 1$ fixed.

load is used and which norm is used to compare coarse-scale and fine-scale solutions. Using the chess-board conductivity problem of Section 6.1, we present numerical evidence that the alternative in which the load is known, and we use a norm that compares errors only on the boundary of the domain is a better choice.

Consider the first choice, that the known local load be used. If we minimize the homogenization error for a particular load, we should get a better solution. This seems obvious. The choice in the case of norm needs a little explanation. As will be shown, if we compare errors only on the boundary (instead of the whole domain) we get a better solution. A finite element interacts with the rest of the domain through the boundary. For example, in

the 1-D example of springs solved in Section 4.1, the process of homogenization reduces the N spring constants to just one. The rest of the domain (or the exterior) cannot know by interacting with an element what is there inside (at least in this 1-D example). In that sense, homogenization is *exact*. But even with the exact homogenization, the errors in the interior are not zero. Thus, they should not be expected to be small, and should not be compared.

Figure 6.6 shows the errors made by making these choices. The curves show the relative errors between k_ϵ and expected exact $k = \sqrt{k_1 k_2}$ while $k_1 = 1$ fixed and k_2 is varied between 1 and 25.

Symbol k_F_SN denotes the curve where the known self-equilibrated "force" (or "load") is used in conjunction with a special norm. Here we use the boundary flux and the special weighs the errors in the interior as 100 times less important than boundary errors. This idea is similar to the weighted ℓ_2 used in the end of Section 4.8. Symbol k_SN denotes the curve where the load is not used, but the special norm is. Symbol k_F_N denotes the curve where the load is used, but the norm is the ℓ_2 norm. Symbol k_N denotes the case where no load is used and the norm is the ℓ_2 norm. As seen, using the known load is better than not using it. Similarly, using the weighted norm is better than not using the weighted norm.

6.3 Computational time for different element sizes

A typical mesh in the sequence of meshes generated for automatic mesh adaptivity will have elements of all sizes. For allocating different elements

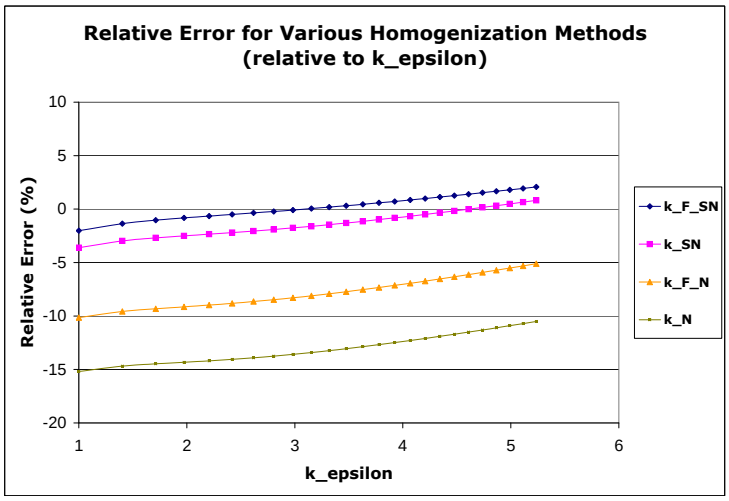


Figure 6.6: The four curves show the relative errors between homogenized conductivity k_ϵ and analytic conductivity $\sqrt{k_1 k_2}$ for various choices made in defining the homogenization error. See text in Section 6.2 for details.

to different processors it is necessary to know in advance the time taken to homogenize elements as a function of element size n (number of SFIL particles in each element side). We compute homogenization time for hexahedral and quadrilateral elements of polynomial degree 1 and identical number of cells in each side.

The most expensive step is computing the pseudoinverse of the element stiffness matrix. In Chapter 5 we presented a few algorithms for computing Moore-Penrose pseudoinverses. Here, we compute pseudoinverses using two kinds of algorithms − sparse Cholesky factorization of (Section 5.4.1) and iterative methods based on known null space (Section 5.4.2).

For the iterative case, we choose the conjugate gradient (CG) and minimum residual (MINRES) methods. As preconditioners, we use Jacobi pre-

conditioner and Incomplete Cholesky (ICC). MINRES is also tried because stiffness matrices in SFIL can be indefinite in a particular iteration. These are implemented in PETSc [11]. The sparse Cholesky factorization is implemented in CHOLMOD [33]. Thus, we analyze four different iterative schemes and one direct scheme. The trends in homogenization times presented below can be taken as trends in times taken to compute pseudoinverse (of the element stiffness matrix) using these 5 different schemes.

Figures 6.7 and Figure 6.8 show the results for 2-D and 3-D respectively. The computation time increases as a power law. For n as the element size, we compute the best-fit homogenization time as a power law $10^A n^B$. The best-fit constants A and B are in Figure 6.9. We mention a few observations

For small elements, sparse Cholesky is better than iterative schemes. In 2-D, it is much better, and never loses to the iterative schemes. In 3-D, for small elements sparse Cholesky is better but not by much. Iterative schemes are better beyond $n = 8$. This might be due to higher fill-in in 3-D.

Between the four iterative schemes, depending on element size n, the best iterative scheme keeps on changing, but they are all very close together. For small n, Jacobi preconditioner is better than ICC for both 2-D and 3-D even though it leads to slower convergence (uses more iterations, not shown here). ICC is more complicated to setup so it does not help much compared to Jacobi unless n increases and the number of iterations taken increase a lot with Jacobi.

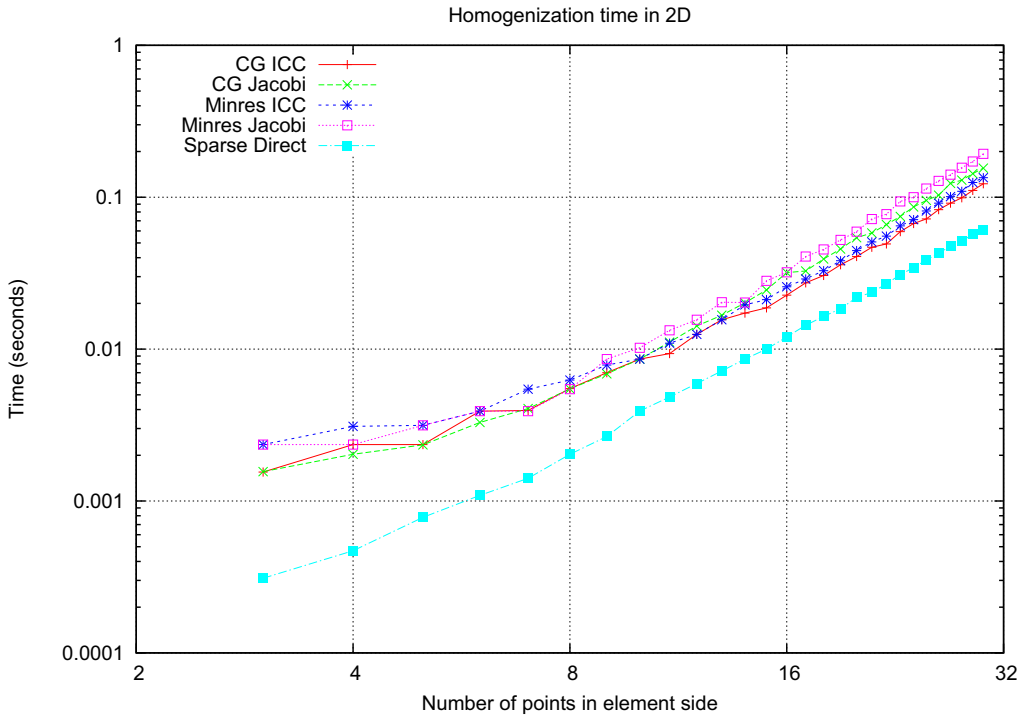


Figure 6.7: The figure shows computation time to homogenize a 2-D quadrilateral element with 8 DOFs (2 per corner) as the element size changes. The element size is number of particles on each side of the quadrilateral. Different pseudoinverse methods lead to different times.

6.4 Optimum element size for homogenization of a cubical lattice

Given a cubical geometry with a fixed number of points p on each side, we might want to divide the lattice into m elements on each side. If there are n points in each element, we have a relation between p, m, and n.

$$p = m\,n - m + 1 \quad \text{or equivalently} \quad m = \frac{p-1}{n-1}$$

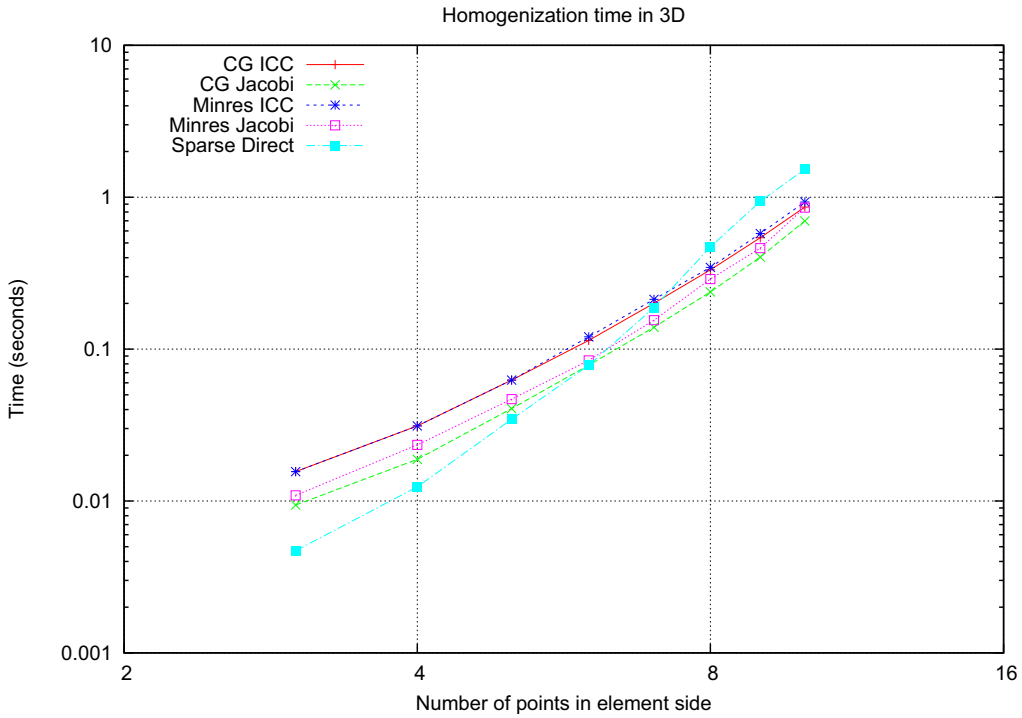


Figure 6.8: The figure shows computation time to homogenize a 3-D hexahedral element with 24 DOFs (3 per corner) as the element size changes. The element size is number of particles on each side of the cube. Different pseudoinverse methods lead to different times.

Figure 6.10 shows a 2-D cartoon of the fine and coarse lattices. We want to decide n to minimize the computational cost of homogenizing the whole cubical lattice into a coarse mesh.

Let $H(n, p)$ be the total time to compute the homogenized Newton step for the full lattice. It consists of m^d homogenizations and 1 linear solve using

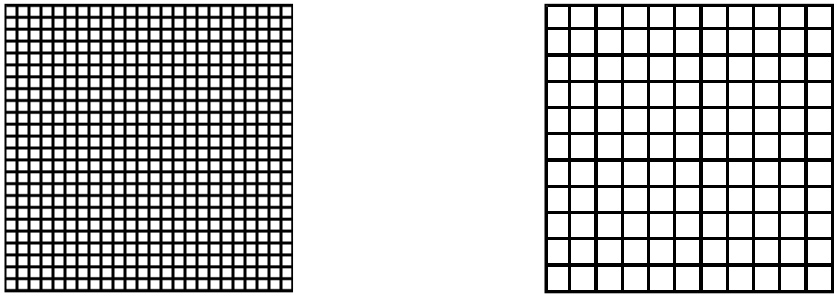
Space dimension = 2

Method	A	B
CG ICC	-4.95	2.73
CG Jacobi	-4.82	2.72
Minres ICC	-4.96	2.77
Minres Jacobi	-4.90	2.82
Cholesky Sparse Direct	-5.09	2.63

Space dimension = 3

Method	A	B
CG ICC	-4.16	4.09
CG Jacobi	-4.68	4.50
Minres ICC	-4.19	4.14
Minres Jacobi	-4.78	4.69
Cholesky Sparse Direct	-5.69	5.91

Figure 6.9: The best-fit power law constants A and B for homogenization time in 2-D (Figure 6.7) and 3-D (Figure 6.8). The computation time is modeled by $T(n) = 10^{-A}n^B$, where n is the element size.



Original lattice

p points on each side

Homogenized lattice

m elements on each side, each with n points

Figure 6.10: A fine-scale lattice with p particles on each side and a coarse mesh (homogenized lattice) with m elements, each containing n point particles.

the homogenized Hessian.

$$H(n,p) = m^d 10^A n^B + \frac{10^{A'}}{d\,2^d}(m+1)^{B'}$$

A, B are best-fit constants related to homogenization of a single element as described in Section 6.3. A', B' are constants for inverting the global homogenized Hessian. They would be different in general. In the analysis ahead, we have chosen $A' = A$ and $B' = B$ so that someone implementing a code can reuse a single scheme for all purposes instead of implementing multiple schemes together.

Substituting $m = \frac{p-1}{n-1}$, we get

$$H(n,p) = \left(\frac{p-1}{n-1}\right)^d 10^A n^B + \frac{10^{A'}}{d\,2^d}\left(\frac{p-1}{n-1}+1\right)^{B'}$$

The objective is to choose an n that minimizes $H(n,p)$ for a given p. If n is 2, we did not homogenize an SFIL lattice really since $m = p-1$. Then, the first term is ignored since it does not require any work. We will spend more time in solving the full system as well as doing more Newton iterations. If $n = p$, it means the lattice is homogenized into a single element. Although the global system will be very small, this will require linear solves for pseudoinverse of a huge matrix. Additionally, the accuracy will be very poor.

We have observed that the optimum element size n depends weakly on changes in p. We will work with $p = 101$. Note that in practice we may choose to not homogenize every element and instead reuse the homogenized Hessian from other equal sized elements. We may also reuse the homogenized Hessian

for multiple Newton steps. This analysis is for a single homogenized Newton step in which all elements are homogenized separately without reusing data from other elements.

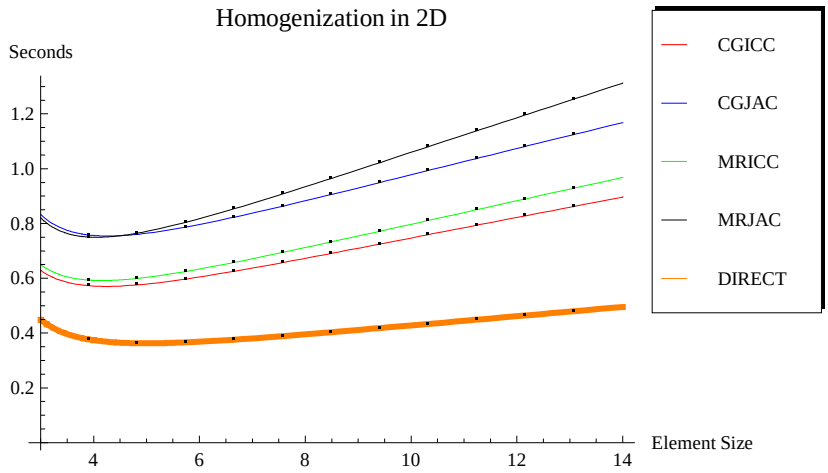


Figure 6.11: The time taken to homogenize a 2-D lattice of size 101×101 once as a function of the coarse element size. Cholesky factorization to compute pseudoinverse is the fastest for all element sizes. Compare Figure 6.12.

Figures 6.11 and Figure 6.12 show the plots for the computation time for a 2-D and 3-D lattice respectively. Each lattice had 101 points on each side. The results are plotted for different methods to compute the pseudoinverses of stiffness matrices. As discussed in Section 6.3, we have analyzed 5 schemes. CG means conjugate gradient, MR means minimum residual, ICC means Incomplete Cholesky preconditioner, JAC means Jacobi preconditioner, and DIRECT means sparse Cholesky factorization.

As we can see, in 3-D, it takes around 4 minutes to homogenize if element size is 4 using sparse Cholesky factorization. For larger 3-D elements,

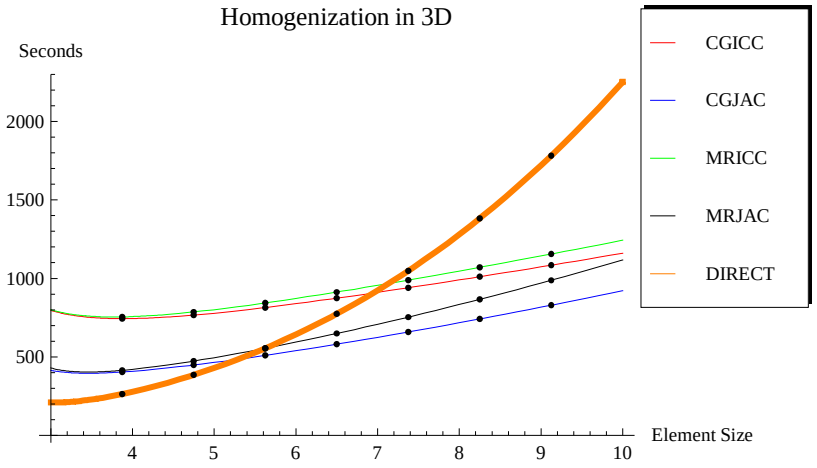


Figure 6.12: The time taken to homogenize a 3-D lattice of size $101 \times 101 \times 101$ once as a function of the coarse element size. Cholesky factorization to compute pseudoinverse is the fastest for small element sizes only. Compare Figure 6.11.

iterative schemes are faster than the sparse direct method. For 2-D, the sparse direct method is always faster. The main reason why a sparse direct solver came out as the best is due to multiple right hand sides. In 3-D, with 3 DOFs per node, 24 (or 25 depending on if we use a known load) linear systems are solved. For the given data, we see that the best idea is to choose $n = 5$ (4 cells on each side), and use a sparse direct solver for local homogenization. Since $n - 1$ is a power of 2, it will be beneficial for mesh adaptivity and refinements. Thus, if initial mesh elements are too large, it is better to break them using the optimum element size as a guideline.

6.5 A comparison of Cholesky factorization and QR factorization for computing pseudoinverse

We discussed sparse direct factorization based algorithms for computing the Moore-Penrose pseudoinverse in Sections 5.4.1 and 5.4.3. The Cholesky factorization algorithm was inherently approximate unlike the QR factorization based algorithm, which was exact. The details of these algorithms are discussed in [33, 34]. We use the stiffness matrix K of a 3-D box with 13 particles on each side (total 2197 particles), and each particle connected to the 26 nearest neighbors. Each particle has 1 DOF. In the ordering, starting from origin, the x direction index is the fastest and z direction index is the slowest.

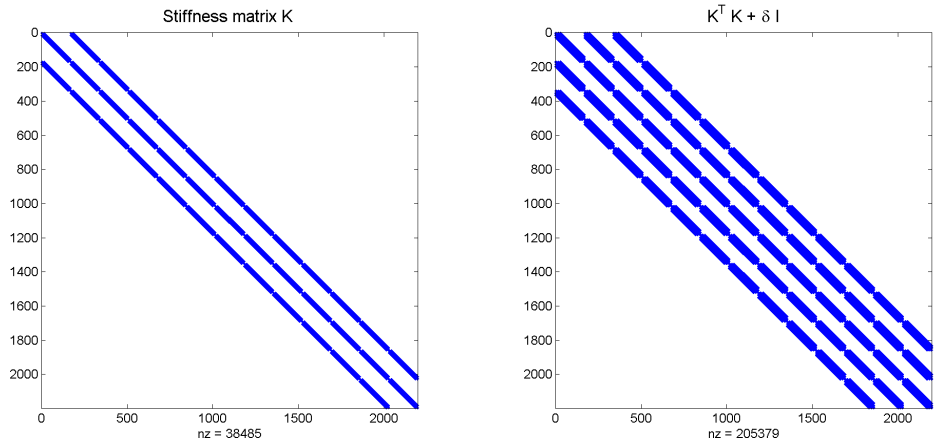
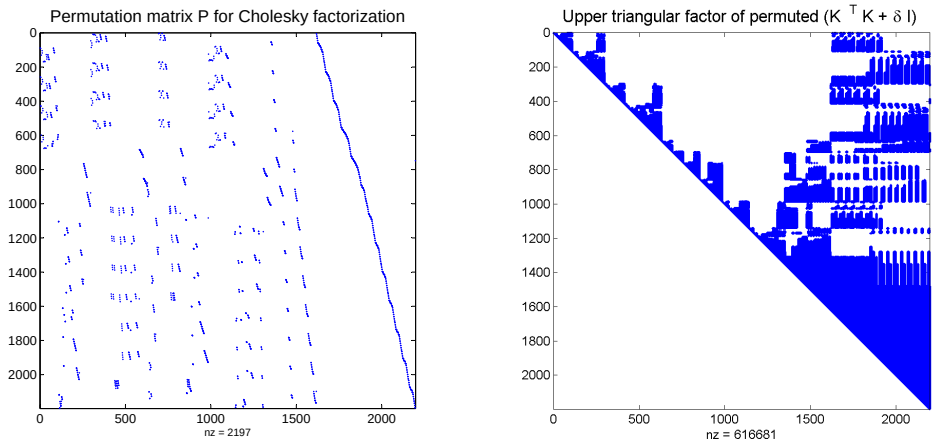


Figure 6.13: (a) The non-zero structure of a stiffness matrix K of a 3-D box with 13 particles on each side (total 2197 particles), and each particle connected to the 26 nearest neighbors. Each particle has 1 DOF. (b) The non-zero structure of $K^T K + \delta I$.

Figures 6.13(a) and (b) show the stiffness matrix and the matrix $K^TK+\delta I$ discussed in Section 5.4.1 used for Tikhonov regularization. Figures 6.14(a) and (b) show the permutation matrix P and the upper triangular factor R such that $P^T(K^TK+\delta I)P = R^TR$. Figure 6.15(a) shows the upper triangular factor R when no permutation is used ($P = I$). The factor has the same bandwidth as the matrix $(K^TK + \delta I)$ and is dense within the band.

Figure 6.15(b) show the permutation (produced by COLAMD [34]) for a fill-in reducing sparse QR factorization of K. Figures 6.16(a) and (b) show the sparse orthogonal factor Q and the sparse upper triangular factor R such that $KP = QR$.



(a) **(b)**

Figure 6.14: (a) The non-zero structure of the permutation matrix P that reduces fill-in for Cholesky factorization of $K^TK + \delta I$. (b) The non-zero structure of the upper triangular Cholesky factor R of $K^TK + \delta I$ such that $P^T(K^TK + \delta I)P = R^TR$.

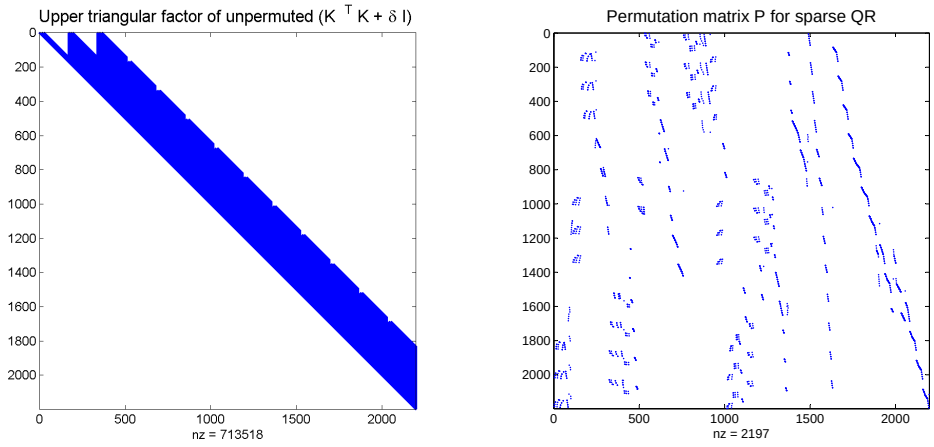


Figure 6.15: (a) The non-zero structure of the upper triangular Cholesky factor R of $K^TK + \delta I$ such that $K^TK + \delta I = R^TR$. No permutation matrix is used. (b) The non-zero structure of the permutation matrix P for a sparse QR decomposition of K such that $KP = QR$. See Figure 6.16.

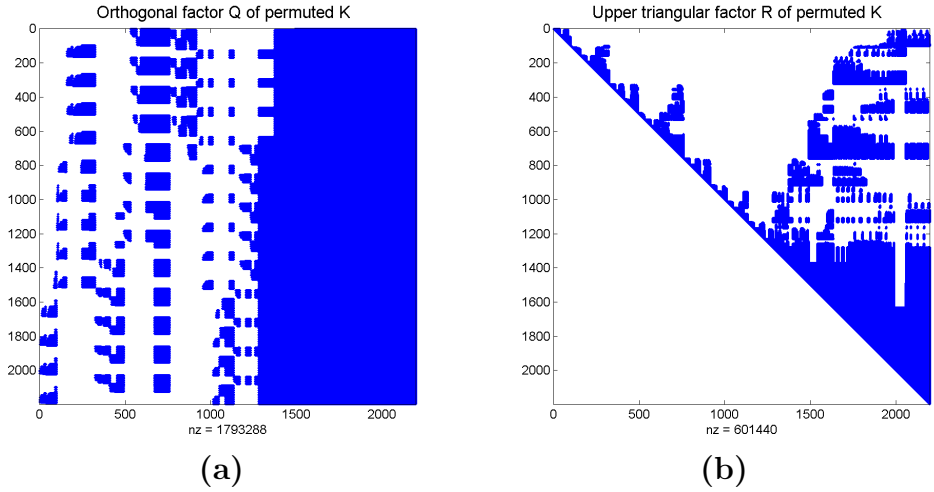


Figure 6.16: (a) The non-zero structure of the orthogonal matrix Q for a sparse QR decomposition of KP. See Figure 6.15(b) for P. (b) The non-zero structure of the upper triangular matrix R where $KP = QR$.

On a 1.5 GHz single core processor, the sparse Cholesky factorization with reordering took 0.235 seconds. The banded Cholesky factorization took 0.238 seconds. The difference is insignificant. However, the upper triangular factor had 16% more non-zeroes for the banded factor. This would make back-substitution slightly expensive.

The sparse QR factorization took 1.196 seconds, nearly 5 times slower than Cholesky factorization. In addition, the number of non-zeroes in Q and R combined was 4 times the number of non-zeroes in the Cholesky factor R.

6.6 Features of sparse algorithms for pseudoinverse

Finally, based on the results presented here and in Sections 6.3 and 6.5, we have created a feature classification matrix for the various pseudoinverse algorithms. This is presented in Figure 6.17. At least for the initial iterations, the approximate pseudoinverse method using Tikhonov regularization presented in Section 5.4.1 may be the best because of its speed.

We have not integrated sparse QR factorization with homogenization yet. The features and classifications are based on analytical formulas and preliminary code testing using [34].

These results are for a specific SFIL lattices, for a single step in the Newton iterations, specific libraries (PETSc, CHOLMOD), and a typical recent CPU (Intel Core 2 Duo, 1.5 GHz). However, for any changes in the configuration, the time analysis can be done automatically for a few typical

Criterion and its importance (to us)	Regularize + Cholesky	Sparse QR	Null space based
Fast (although this is element dependent)	Fast	Good	Depends (If multiple RHS)
Multiple RHS	Factorize once	Factorize once	Multiple solves
Software complexity	Medium	High	Medium
Exact (up to floating point)	No	Yes	Depends (iterative)
Prior information needed	Smallest singular value magnitude	No	Null space
Conditioning	Bad (little better if Sparse QR used)	Good	Good

Figure 6.17: Features of algorithms for computing Moore-Penrose pseudoinverse discussed in Chapter 5. None of the algorithms is the best amongst all depending on the kind of problem to be homogenized.

elements and the best method can be chosen accordingly at run-time.

6.7 Convergence rate of uniform mesh refinements

We present the results of homogenization of a 2-D lattice with 21 particles on each side and "random" bonds. The bottom layer of particles is fixed with equal inter-particle distance 1.3 and the lattice relaxes under zero external forces. The objective is to compute the convergence rate of error in energy norm as the coarse mesh element size is reduced to the fine-scale elements.

The steps of homogenization and nonlinear Newton iterations are integrated as shown earlier in Figure 4.6. In this case, however, the mesh is always refined uniformly in h.

Figure 6.18 shows the lattice topology. The lattice consists of two kinds of harmonic bonds with different parameters (equilibrium length and stiffness). The bonds are randomly distributed. The darker color represents the first kind of bond with stiffness 0.4 and length 1.2. The lighter color signifies the second kind with stiffness 1 and length 1. Figure 6.19 shows an equilibrium solution starting from a square lattice as the initial guess.

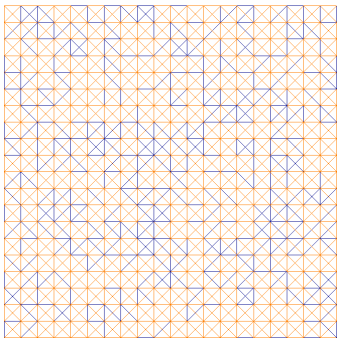


Figure 6.18: A "random" 2-D lattice with 21 particles on each side.

We use various element sizes for the coarse mesh to homogenize the lattice. Figure 6.21 shows the results. The nonlinear problem is homogenized using load-independent homogenization (Section 4.4) with 1, 4, and 16 "square" elements with 400, 100, and 25 lattice "cells" in each element. As seen, the solutions obtained after homogenization match the fine-scale solution very well. Figure 6.20 shows the global relative error in ℓ^2 norm when compared to the non-homogenized fine-scale solution.

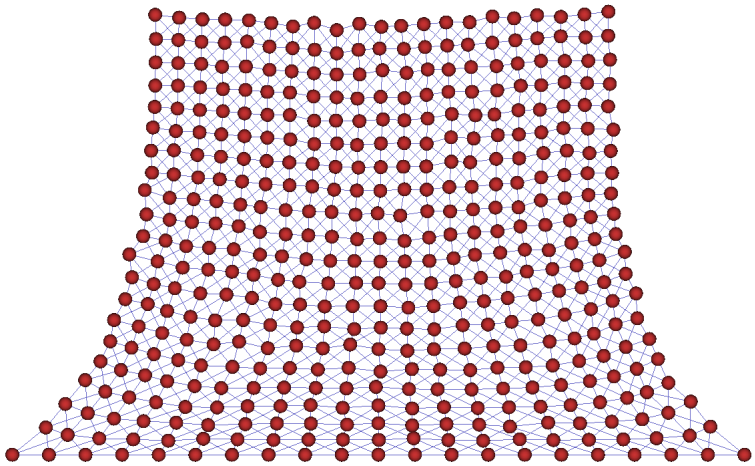


Figure 6.19: An equilibrium solution for the lattice topology shown in Figure 6.18 and fixed bottom layer.

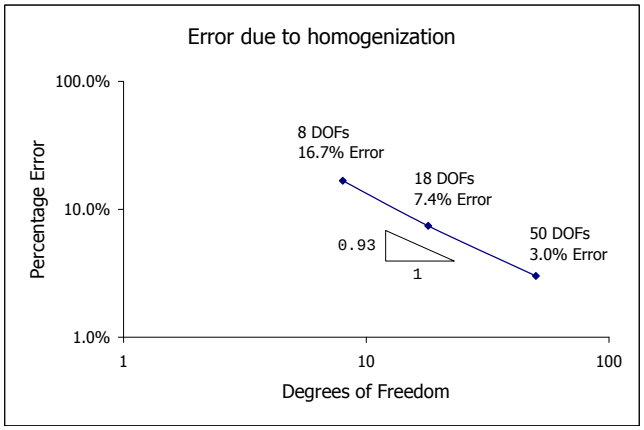


Figure 6.20: The global relative error in ℓ^2 norm for various levels of homogenization shown in Figure 6.21. The exact solution is specified by 882 degrees of freedom.

6.8 Integration of homogenization with adaptivity and Newton iterations

In this section, we show results that combine automatic h-adaptivity with nonlinear iterations and local homogenization for SFIL lattice problems.

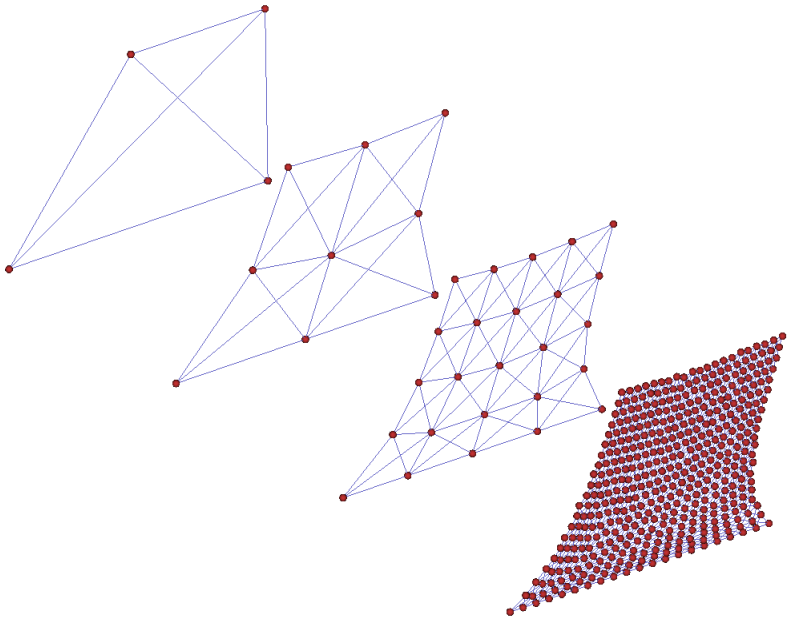


Figure 6.21: An oblique view of the equilibrium solutions for the lattice shown in Figure 6.18 after homogenizing the lattice at various resolutions. The bottom-right lattice is the non-homogenized solution taken from Figure 6.19.

These three techniques are integrated by the logic shown in Figure 4.6. We present results for 2-D and 3-D lattices and compare meshes generated by energy-oriented and goal-oriented adaptivity. In contrast, the results presented in Chapter 3 were for automatic *hp*-adaptivity for linear problems with smooth material data on cubic lattices and without local homogenization.

Although the real SFIL lattice model is in 3-D, we show the progress of mesh refinements in 2-D first because of the clarity it provides. This 2-D lattice model has 128×128 cells. The base is fixed and the lattice is deformed to its minimum energy configuration for a sequence of coarse meshes. We run the algorithm two times but use a different strategy in both cases. First we

refine solely to minimize error in energy. In the second case the refinements are to minimize an error estimate in a goal functional. As shown in Figure 6.22, the goal is the distance between top-right corner and the center of top edge. As shown in Figures 6.24(a) and 6.28(a), the initial mesh consists of 4 equal-sized elements. The analysis and results are presented in Sections 6.8.1 and 6.8.2.

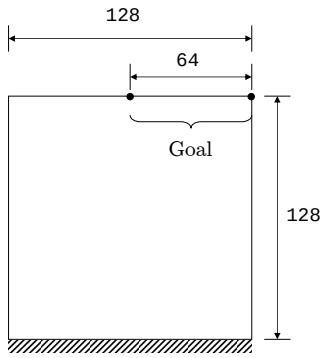


Figure 6.22: The figure shows a 2-D model with 16384 unit SFIL cells and fixed base. For goal-oriented adaptivity, the goal is the distance between top-right corner and the center of top edge. The numerical results for this model are shown in Figures $6.24 - 6.30$.

The 3-D model, shown in Figure 6.23, depicts a SFIL lattice with rectangular features for imprinting. It consists of approximately 13000 unit cells and 38000 DOFs. The goal is the distance between the two vertical blocks measured. This distance is measured between coordinates (12, 28, 4) and (12, 24, 4). The initial mesh consists of equal-sized elements each of edge size 4, shown in Figures 6.34(a). See Sections 6.8.3 and 6.8.4 for analysis and results.

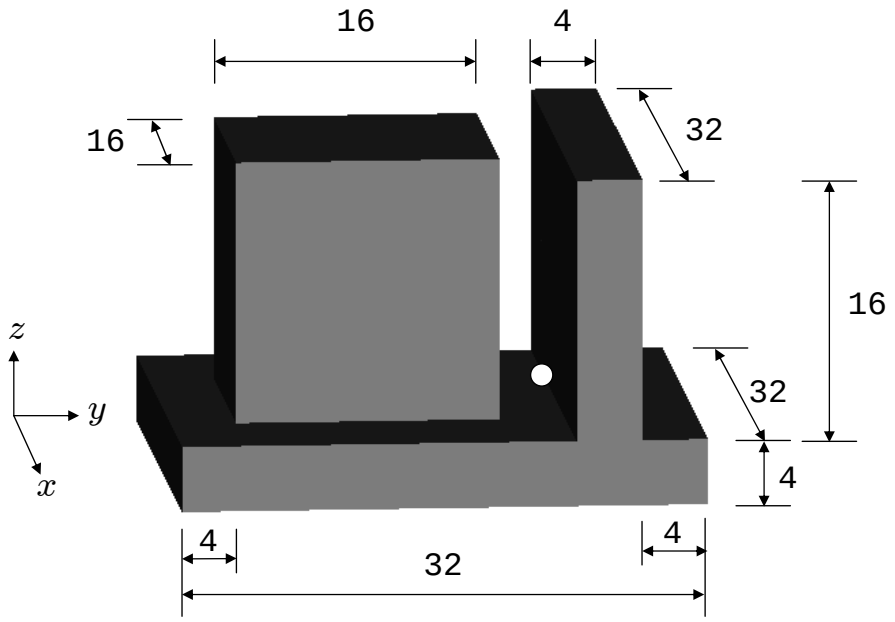
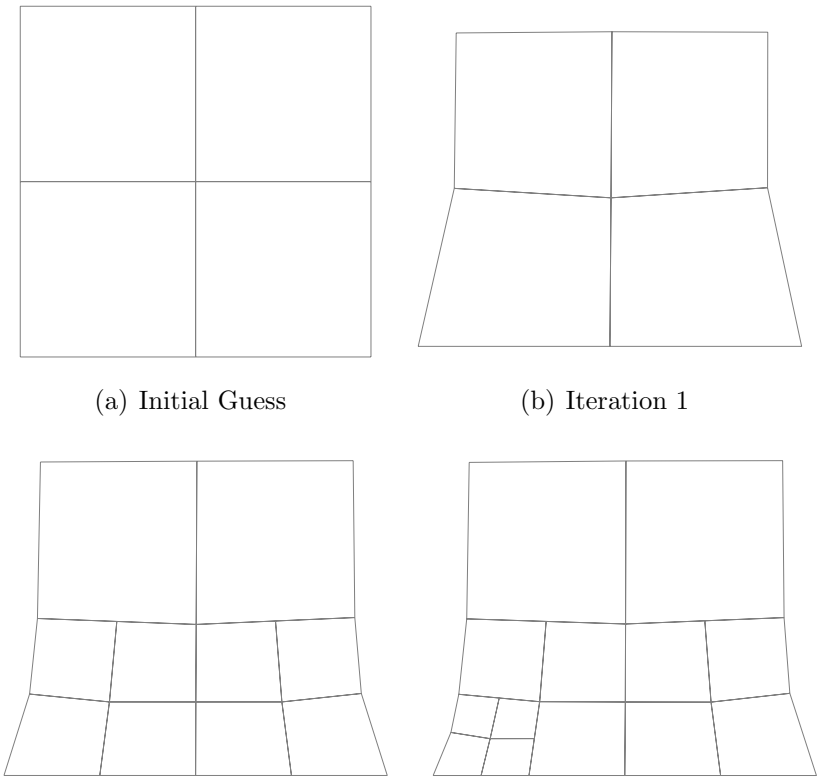


Figure 6.23: The figure shows a 3-D model with approximately 13000 unit SFIL cells and fixed base. For goal-oriented adaptivity, the goal is the distance between coordinates (12, 28, 4), shown by the white circle, and (12, 24, 4). The numerical results for this model are shown in Figures 6.34 – 6.42.

6.8.1 Energy-oriented mesh adaptivity for a 2-D mesh

We show energy-oriented adaptivity results for the 2-D model shown in Figure 6.22. Figures 6.24 – 6.25 show the mesh as refinements proceed. Meshes are shown after the Newton iterations for that particular mesh have converged. As seen, the mesh refinements proceed to have smaller elements on the base, specifically near the two corners. This is expected because of a change of boundary condition – free "boundary" on vertical edges and fixed "boundary" on the base. By boundary we mean the molecules that form the topological boundary of the lattice in analogy with a surface in continuum

material. The refinements are not necessarily symmetric in each step because different bonds and molecules are randomly distributed, being generated by the Monte-Carlo algorithm discussed in Section 2.2.3.



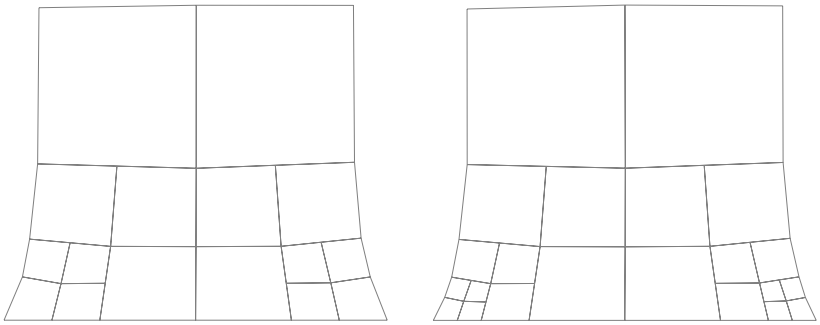
(a) Initial Guess

(b) Iteration 1

(c) Iteration 2

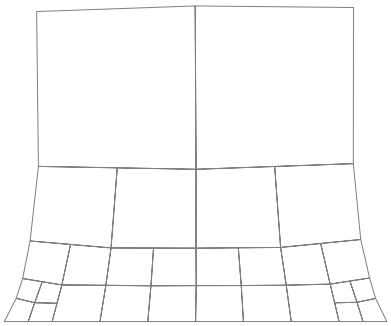
(d) Iteration 3

Figure 6.24: Energy-oriented mesh adaptivity for a 2-D lattice of size 128 $\times$ 128 with fixed base boundary condition. Rest of the iterations are shown in Figure 6.25. Compare goal-oriented mesh adaptivity in Figures $6.28 - 6.29$.

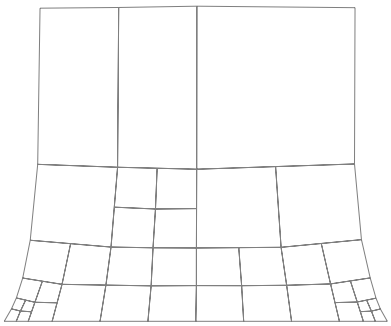


(a) Iteration 4

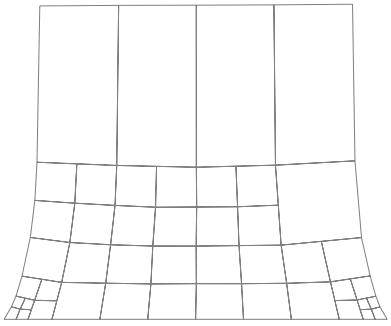
(b) Iteration 5



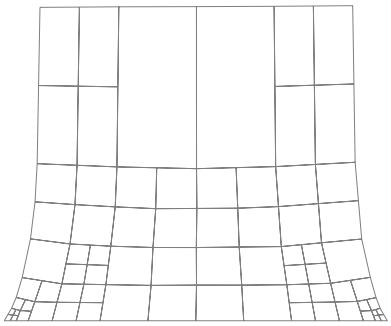
(c) Iteration 6



(d) Iteration 7

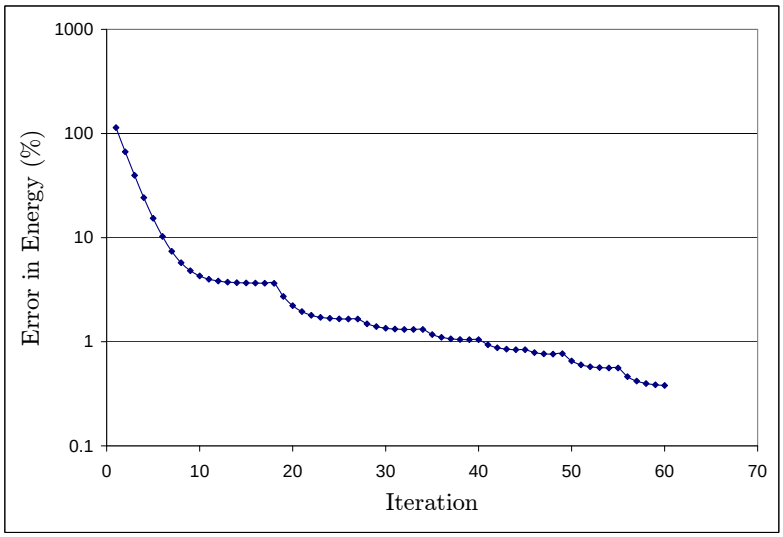


(e) Iteration 8

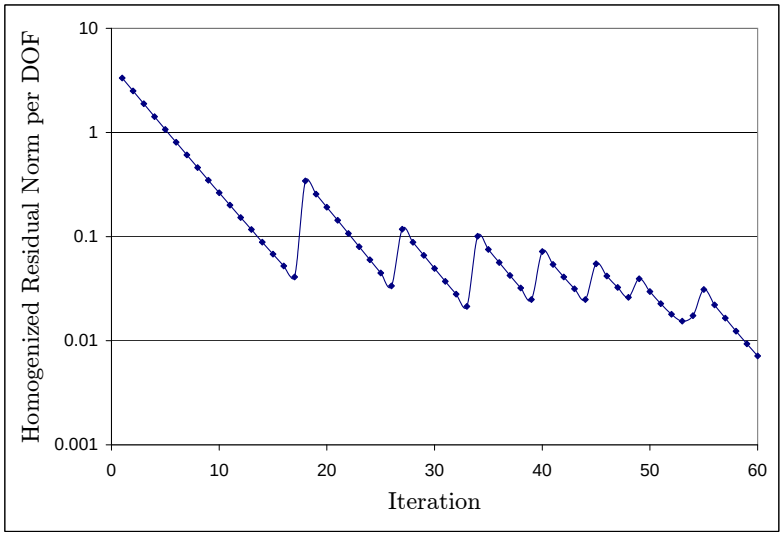


(f) Iteration 9

Figure 6.25: Energy-oriented mesh adaptivity for a 2-D lattice of size 128 $\times$ 128 with fixed base boundary condition. Previous iterations are shown in Figure 6.24. Compare goal-oriented mesh adaptivity in Figures $6.28 - 6.29$.

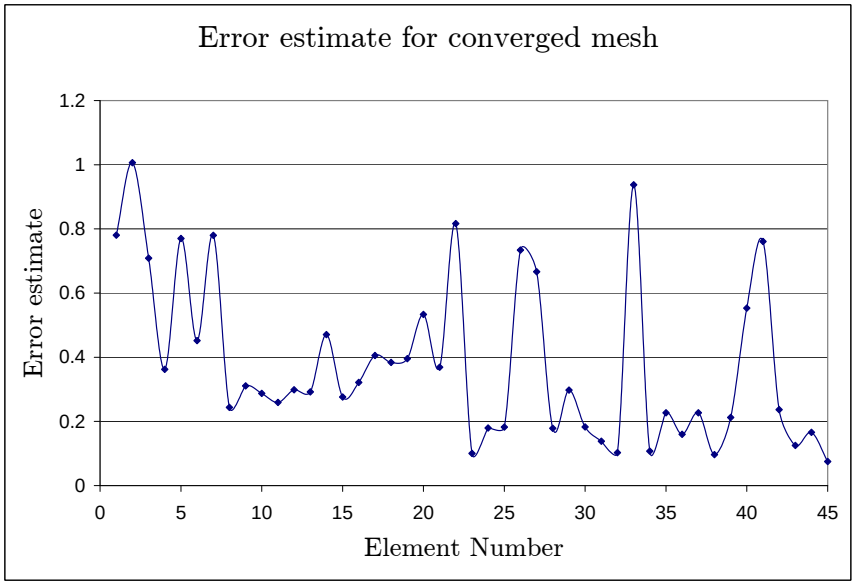


(a) Error in energy (%)

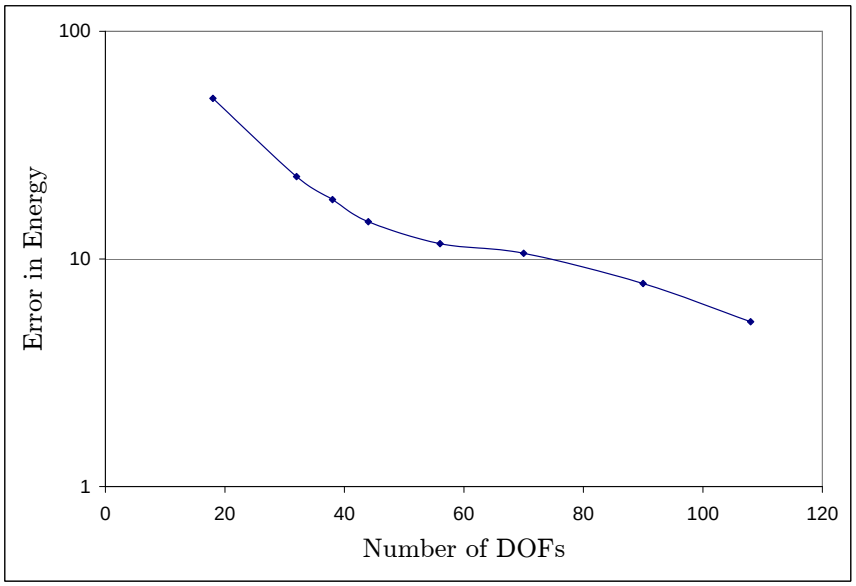


(b) Homogenized residual norm per DOF

Figure 6.26: The graphs show how the minimum energy and residual norm change as mesh is refined using energy-oriented adaptivity and Newton iterations proceed. (a) The kinks are present where mesh is refined and the instantaneous rate of decrease of energy is higher. (b) The residual decreases with Newton iterations and goes up again upon mesh refinements. This is because fine-scale becomes more important. Compare with Figure 6.31.



(a) *A posteriori* error estimates in various elements



(b) Error in energy versus number of DOFs

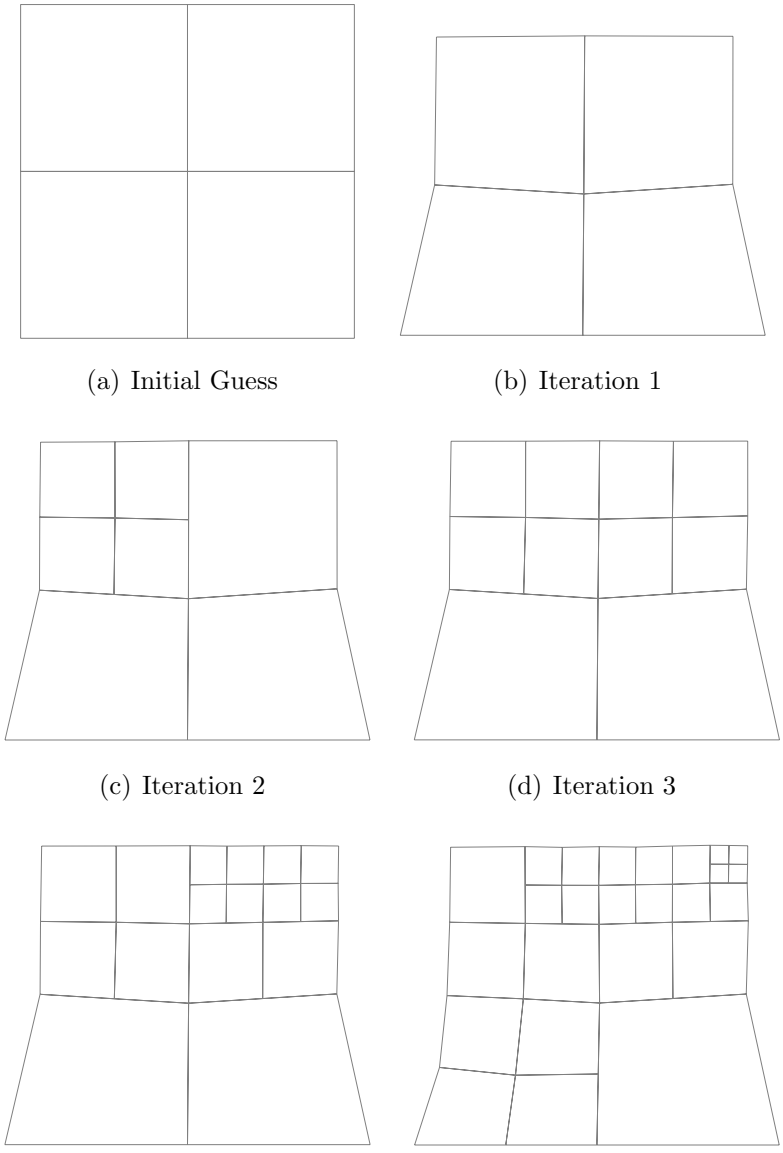
Figure 6.27: The graphs show the distribution of errors in various elements and the decrease of error in energy with the increasing number of DOFs for energy-oriented adaptivity. Compare with Figure 6.32.

Figure 6.26(a) shows convergence of error, measured as the difference of energy for current mesh and the minimum energy, as Newton iterations and mesh iterations proceed. The kinks are at iterations where the mesh is refined because the solution has numerically converged for the current mesh. Except while approaching the kinks (from the left) the reduction in error is smooth. In Figure 6.26(b), we plot the homogenized residual per DOF at each iteration. By homogenized residual we mean the vector that drives the Newton iterations. We divide by the number of DOFs because the number of DOFs increases as the mesh is refined. The residual goes up again upon mesh refinements because further details of the fine-scale are seen. The solution is the best in energy only for the coarser mesh.

Figure 6.27(a) shows *a posteriori* error estimates in various elements for the last mesh. The variation in error in different elements is not too large, which is a good sign. Figure 6.27(b) is a plot of absolute error versus number of DOFs as the refinements proceed. The rate of decrease is almost constant throughout the iterations. Compare this with the plot in Figure 6.32 for goal-oriented adaptivity in which the corresponding curve is nearly flat.

6.8.2 Goal-oriented mesh adaptivity for a 2-D mesh

We show goal-oriented adaptivity results for the 2-D model shown in Figure 6.22. Figures $6.28 - 6.29$ show the mesh as refinements proceed. As expected, the mesh refinements proceed to have smaller elements near the two points that are needed to compute the goal.

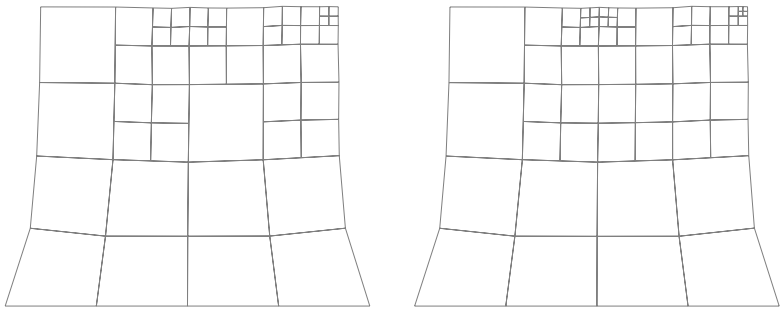


(a) Initial Guess (b) Iteration 1

(c) Iteration 2 (d) Iteration 3

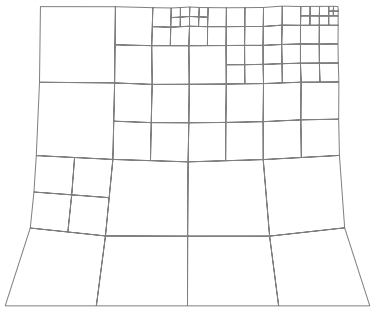
(e) Iteration 4 (f) Iteration 5

Figure 6.28: Goal-oriented mesh adaptivity for a 2-D lattice of size 128 $\times$ 128 with fixed base boundary condition. The goal is the distance between the top-right corner and center of top edge. Rest of the iterations are shown in Figure 6.29. Compare energy-oriented mesh adaptivity in Figures $6.24 - 6.25$.

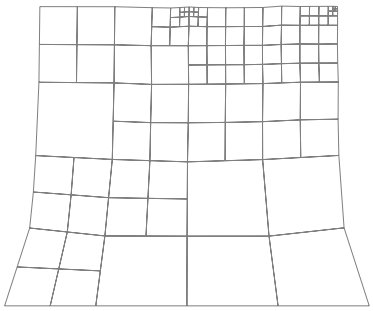


(a) Iteration 6

(b) Iteration 7



(c) Iteration 8



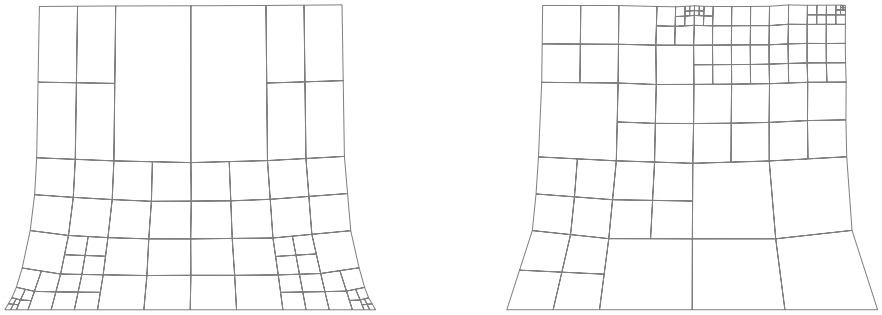
(d) Iteration 9

Figure 6.29: Goal-oriented mesh adaptivity for a 2-D lattice of size 128 $\times$ 128 with fixed base boundary condition. The goal is the distance between the top-right corner and center of top edge. Previous iterations are shown in Figure 6.28. Compare energy-oriented mesh adaptivity in Figures $6.24 - 6.25$.

Figure 6.30 shows the meshes for the converged solution for energy-oriented and goal-oriented adaptivity. Figure 6.31(a) shows convergence of error, measured as the difference of energy for current mesh and the minimum energy, as Newton iterations and mesh iterations proceed. The kinks are at some of the iterations where the mesh is refined because the solution has numerically converged for the current mesh. The curve is much flatter than the curve shown in Figure 6.26(a) because the primary purpose of iterations

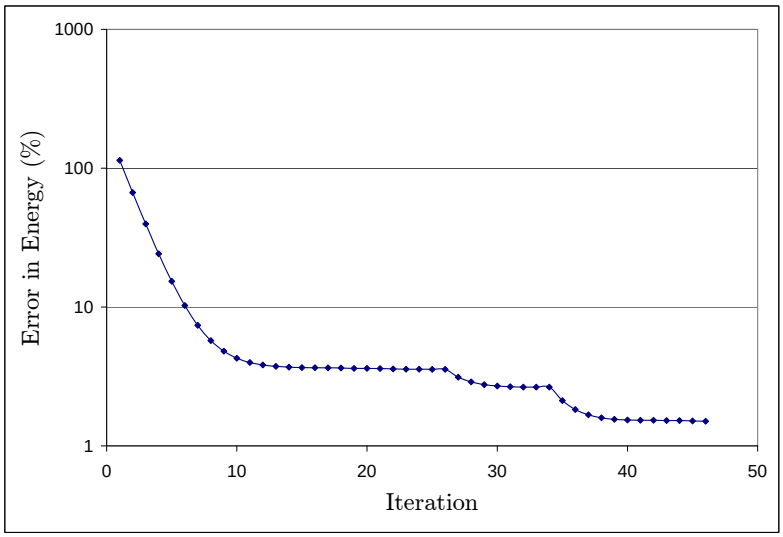
is to reduce the error in goal. In Figure 6.31(b), we plot the homogenized residual per DOF at each iteration. By homogenized residual we mean the vector that drives the Newton iterations. We divide by the number of DOFs because as the mesh is refined the number of DOFs increases. The residual goes up again upon mesh refinements because further details of the fine-scale are seen. The solution is the best in energy only for the coarser mesh.

Figure 6.32(a) shows *a posteriori* error estimates in various elements for the last mesh. The variation in error in different elements is not too large, which is a good sign. Figure 6.32(b) is a plot of absolute error versus number of DOFs as the refinements proceed. The error is almost constant throughout the iterations. Compare this with the plot in Figure 6.27(b) for energy-oriented adaptivity in which the corresponding curve does not have flat regions.

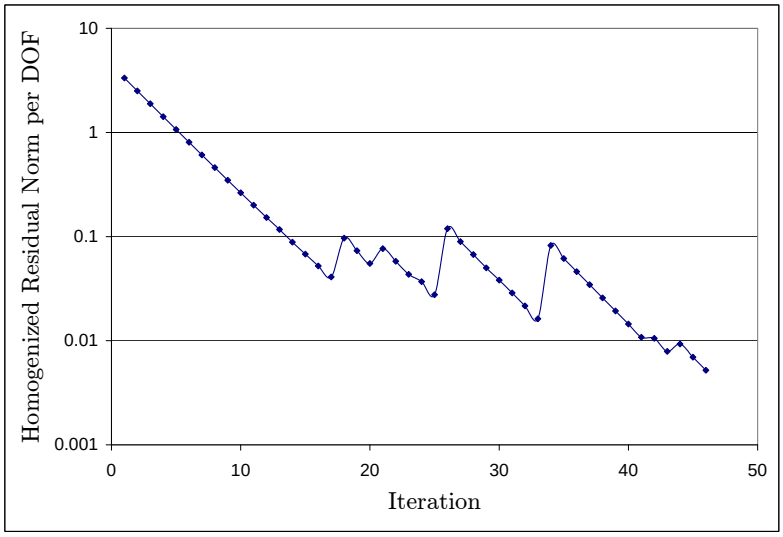


(a) Energy-oriented adaptivity (b) Goal-oriented adaptivity

Figure 6.30: Final meshes generated by energy-oriented adaptivity and goal-oriented adaptivity. The goal is the distance between top-right corner and the center of top edge.

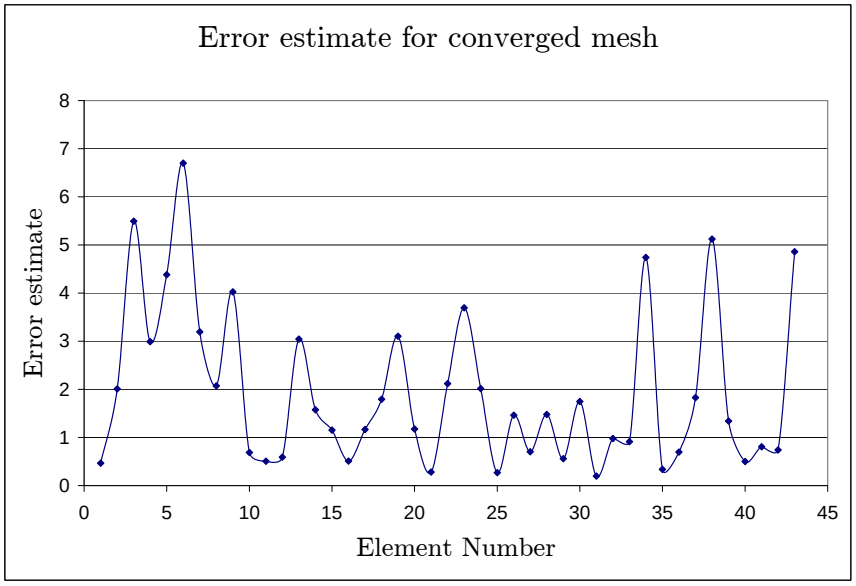


(a) Error in energy (%)

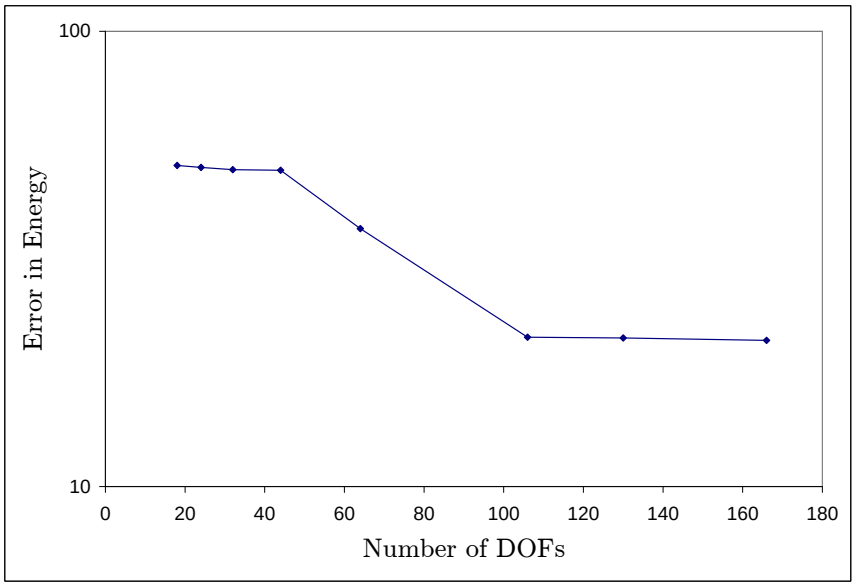


(b) Homogenized residual norm per DOF

Figure 6.31: The graphs show how the minimum energy and residual norm change as mesh is refined using goal-oriented adaptivity and Newton iterations proceed. (a) The kinks are present where mesh is refined and the instantaneous rate of decrease of energy is higher. (b) The residual decreases with Newton iterations and goes up again upon mesh refinements. This is because fine-scale becomes more important. Compare with Figure 6.26.

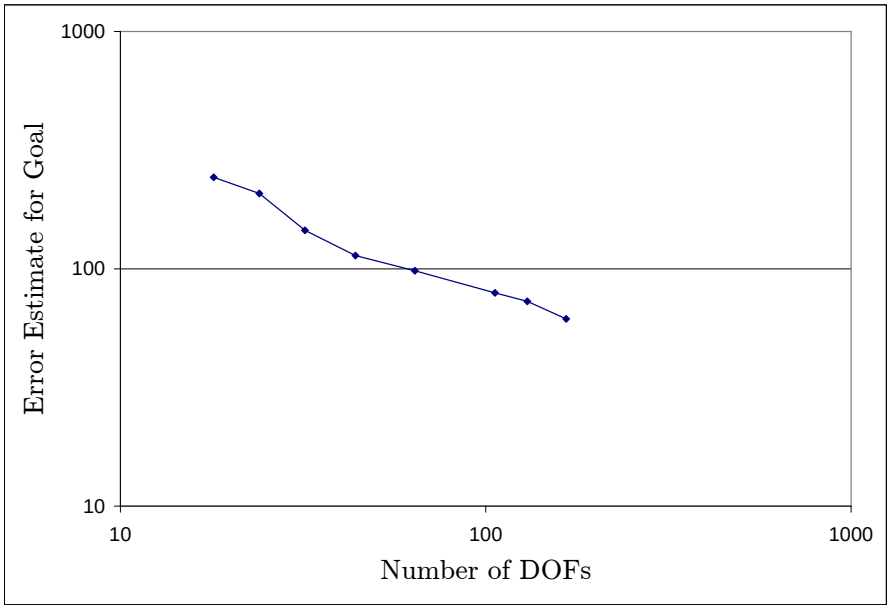


(a) A posteriori error estimates in various elements

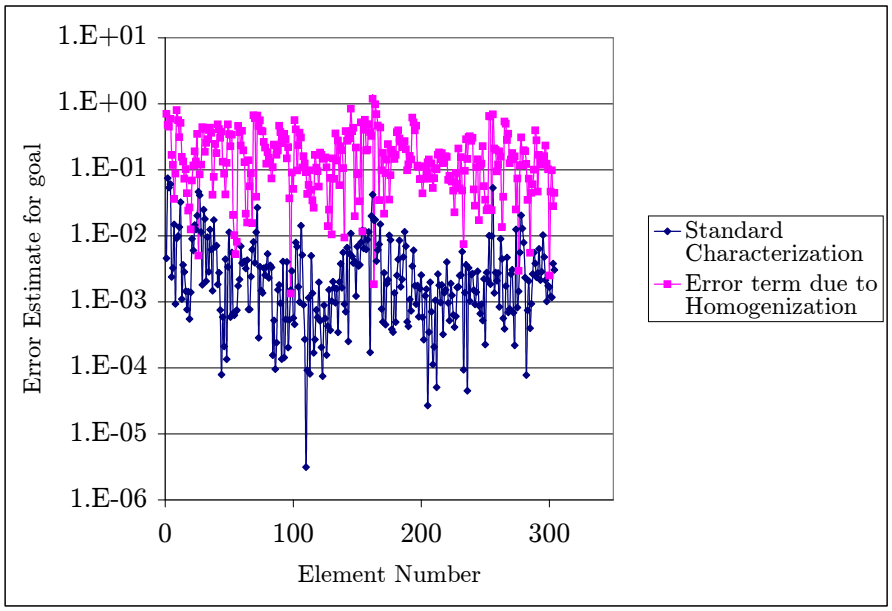


(b) Error in energy versus number of DOFs

Figure 6.32: The graphs show the distribution of errors in various elements and the decrease of error in energy with the increasing number of DOFs for goal-oriented adaptivity. Compare with Figure 6.27.



(a) Error estimate in goal versus number of DOFs



(b) Components of error estimate as discussed in Section 4.10

Figure 6.33: The decrease in error estimate for goal-oriented mesh adaptivity and components of error estimate as discussed in Section 4.10.

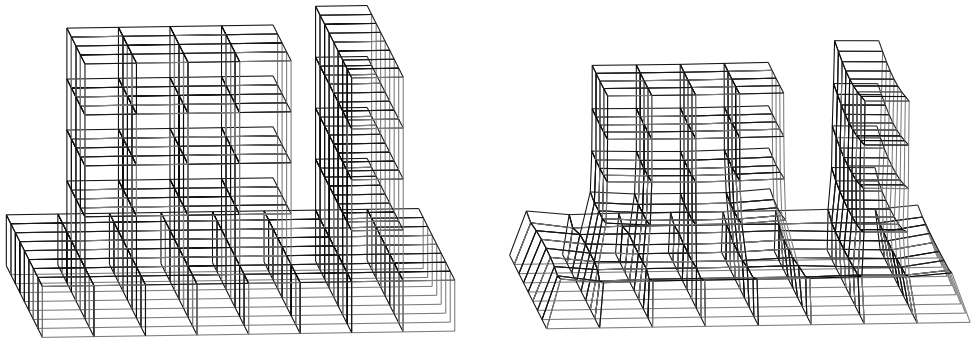
The error estimate in goal, however, decreases uniformly as seen in Figure 6.33(a). We also plot the error estimates for individual elements in Figure 6.33(b). The estimate is divided into two parts − the standard characterization and the term due to homogenization (Section 4.10). The error due to homogenization is larger by around 1 order of magnitude.

6.8.3 Energy-oriented mesh adaptivity for a 3-D mesh

We show energy-oriented adaptivity results for the 3-D model shown in Figure 6.23. Figures $6.34 - 6.35$ show the mesh as refinements proceed. Meshes are shown after the Newton iterations for that particular mesh have converged. We start with a mesh that has equal elements of size 4. As seen, the mesh refinements proceed to have smaller elements where the lattice has abrupt changes, either in boundary conditions or the presence of topological features like corners or edges. The first refinements just take care of the four corners of the base (Figure 6.35(a)). Next come the four edges of the base and the 4 corners of the block placed on the base (Figure 6.35(b)). Similar to the 2-D model in Figures $6.24 - 6.25$, the refinements happen near discontinuities in the boundary and boundary conditions. The final mesh is shown more clearly in Figure 6.36 with the view axis aligned to the coordinate axes.

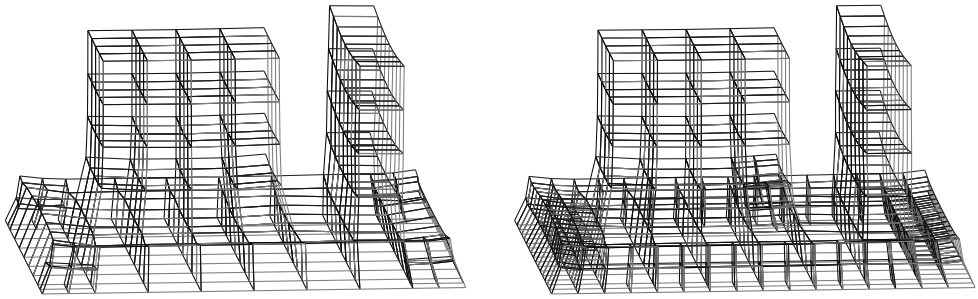
Figure 6.37(a) shows convergence of error, measured as the difference of energy for current mesh and the minimum energy, as Newton iterations and mesh iterations proceed. The kinks are at iterations where the mesh is refined because the solution has numerically converged for the current mesh.

Except while approaching the kinks (from the left) the reduction in error is smooth. In Figure 6.37(b), we plot the homogenized residual per DOF at each iteration. By homogenized residual we mean the vector that drives the Newton iterations. We divide by the number of DOFs because as the mesh is refined the number of DOFs increases. The residual goes up again upon mesh refinements because further details of the fine-scale are seen. The solution is the best in energy only for the coarser mesh. Figure 6.38(a) shows *a posteriori* error estimates in various elements for the last mesh. The elements with lower number are the initial mesh elements. Some of them have not been refined yet and this explains larger error in the left of the graph. Figure 6.38(b) is a plot of absolute error versus number of DOFs as the refinements proceed. The rate of decrease is almost constant throughout the iterations. Compare this with the almost flat curve in Figure 6.44 for goal-oriented adaptivity.



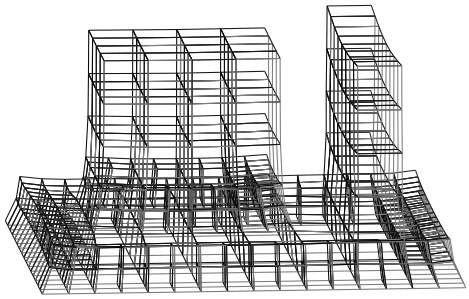
(a) Initial Guess (b) Iteration 1

Figure 6.34: Energy-oriented mesh adaptivity for a 3-D lattice shown in Figure 6.23 with fixed base boundary condition. Rest of the iterations are shown in Figure 6.35. Compare goal-oriented mesh adaptivity in Figures $6.39 - 6.40$.

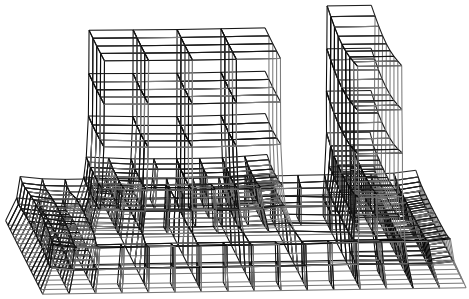


(a) Iteration 2

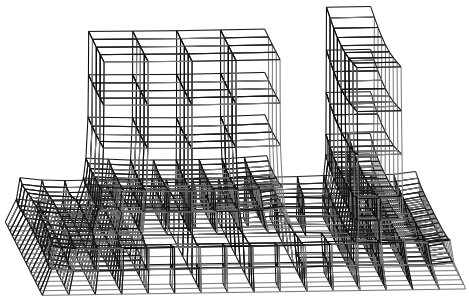
(b) Iteration 3



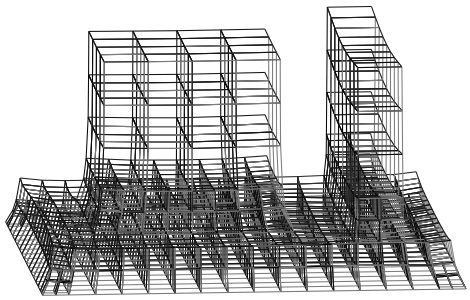
(c) Iteration 4



(d) Iteration 5

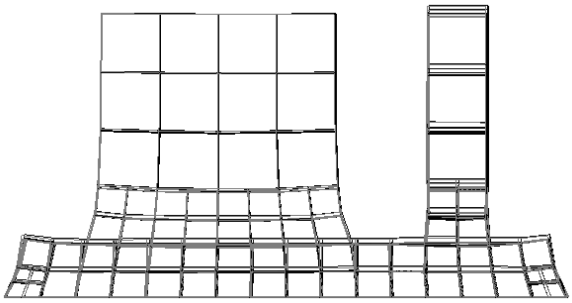


(e) Iteration 6

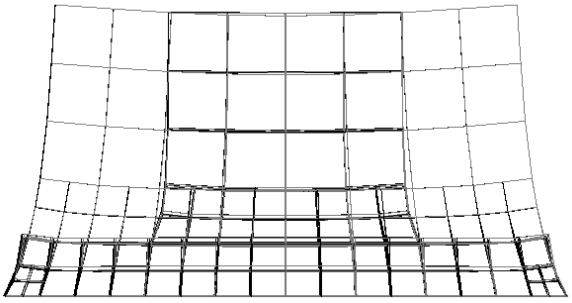


(f) Iteration 7

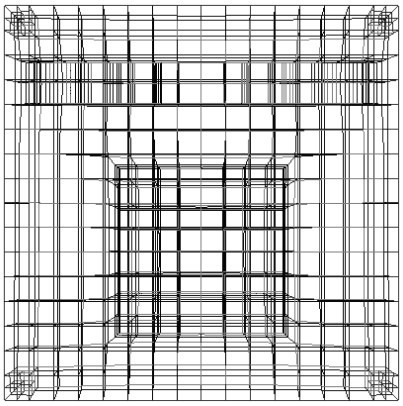
Figure 6.35: Energy-oriented mesh adaptivity for a 3-D lattice shown in Figure 6.23 with fixed base boundary condition. Previous iterations are shown in Figure 6.34. Compare goal-oriented mesh adaptivity in Figures $6.39 - 6.40$.



(a) Front view

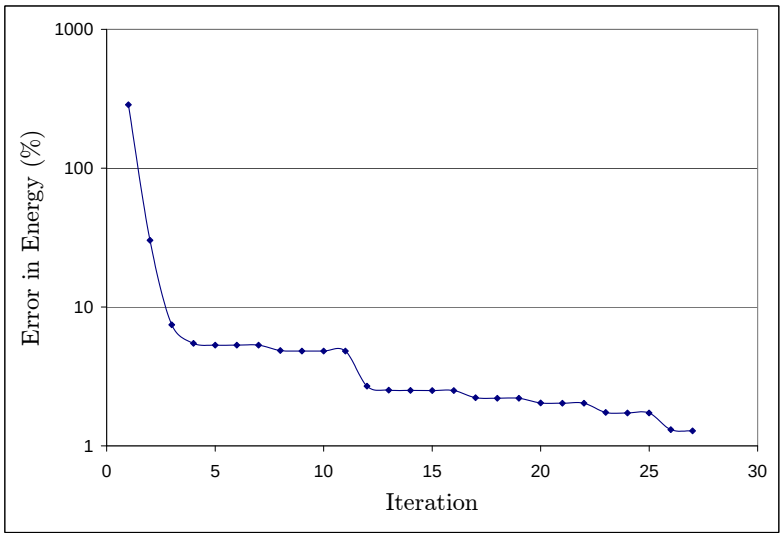


(b) Side view

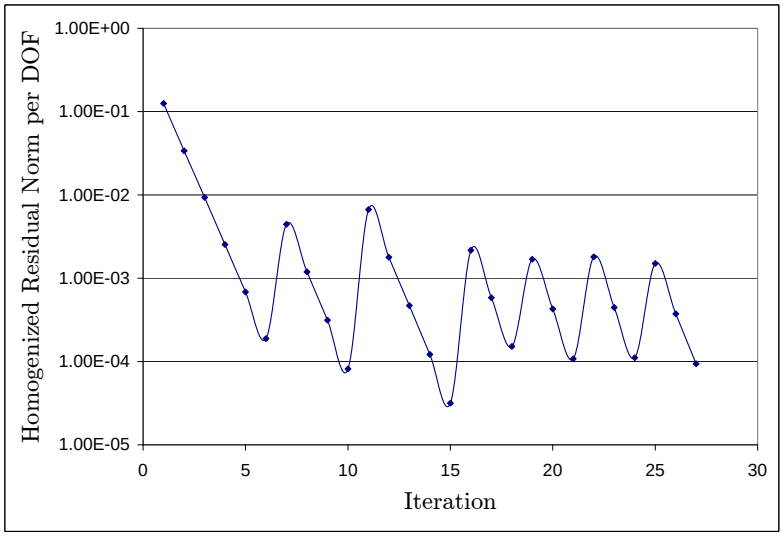


(c) Top view

Figure 6.36: Front, side, and top views of converged mesh generated by energy-oriented adaptivity iterations shown in Figures 6.34 and 6.35. Compare with Figure 6.41.

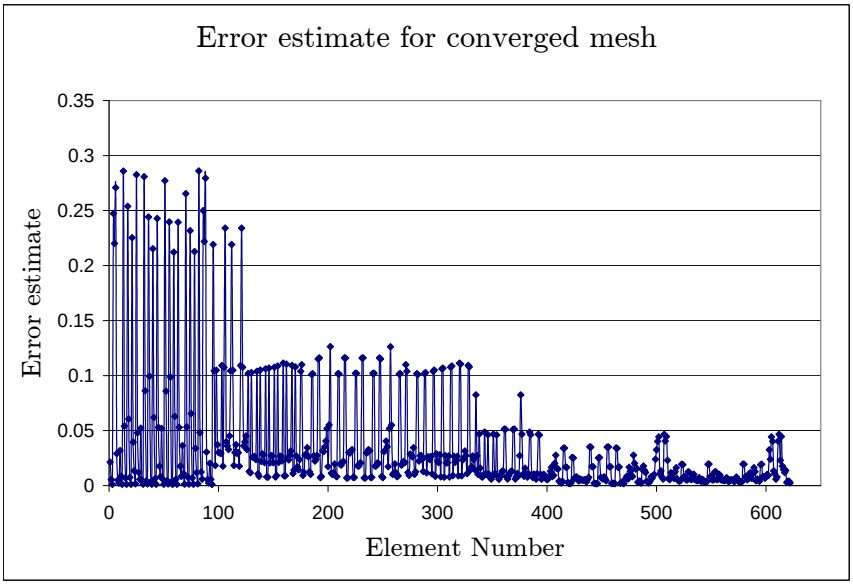


(a) Error in energy (%)

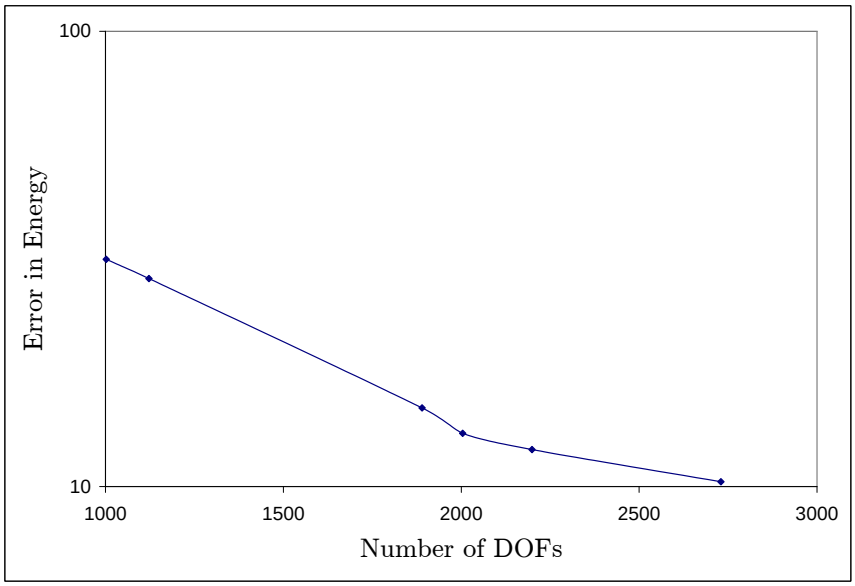


(b) Homogenized residual norm per DOF

Figure 6.37: The graphs show how the minimum energy and residual norm change as mesh is refined using energy-oriented adaptivity and Newton iterations proceed. (a) The kinks are present where mesh is refined and the instantaneous rate of decrease of energy is higher. (b) The residual decreases with Newton iterations and goes up again upon mesh refinements. This is because fine-scale becomes more important. Compare with Figure 6.43.



(a) A posteriori error estimates in various elements

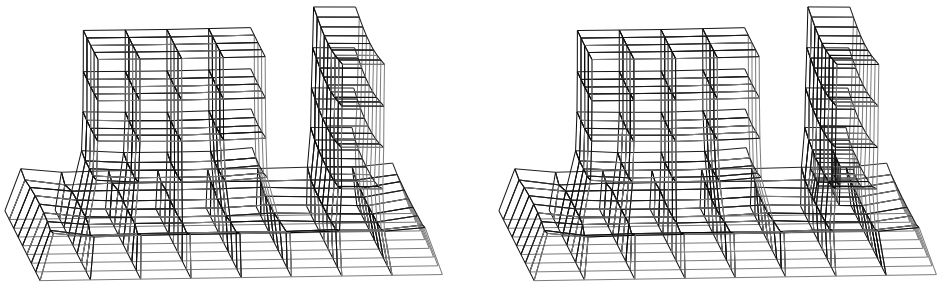


(b) Error in energy versus number of DOFs

Figure 6.38: The graphs show the distribution of errors in various elements and the decrease of error in energy with the increasing number of DOFs for energy-oriented adaptivity. Compare with Figure 6.44.

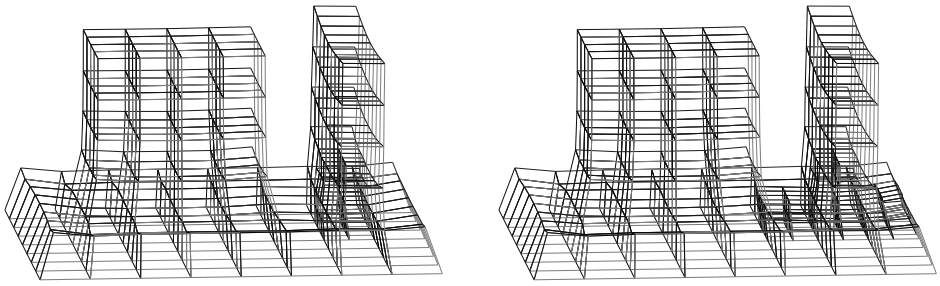
6.8.4 Goal-oriented mesh adaptivity for a 3-D mesh

We show energy-oriented adaptivity results for the 3-D model shown in Figure 6.23. Figures $6.39 - 6.40$ show the mesh as refinements proceed. Meshes are shown after the Newton iterations for that particular mesh have converged. We start with a mesh that has equal elements of size 4. The goal is the distance between coordinates (12, 28, 4) and (12, 24, 4), see Figure 6.23. The initial guess is shown in Figure 6.34(a). As expected, the mesh refinements proceed to have smaller elements near these two points. Most of the larger elements away from these two points are not refined, even though they need to be refined for energy-oriented adaptivity (Figures $6.34 - 6.35$). The final mesh is shown more clearly in Figure 6.41 with the view axis aligned to the coordinate axes.



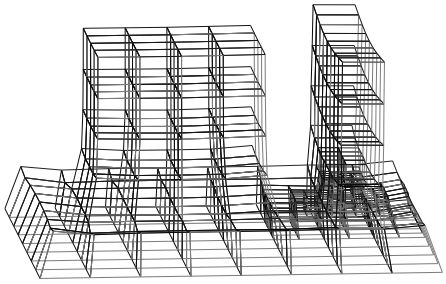
(a) Iteration 1 (b) Iteration 2

Figure 6.39: Goal-oriented mesh adaptivity for a 3-D lattice shown in Figure 6.23 with fixed base boundary condition. The goal is the distance between the two blocks measured near the bottom. Rest of the iterations are shown in Figure 6.40. Compare energy-oriented mesh adaptivity in Figures $6.34 - 6.35$.

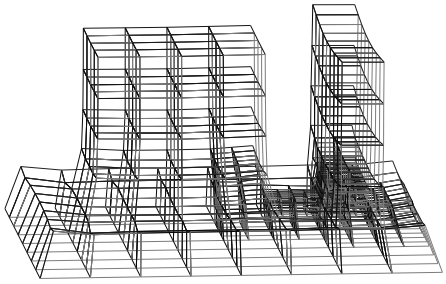


(a) Iteration 3

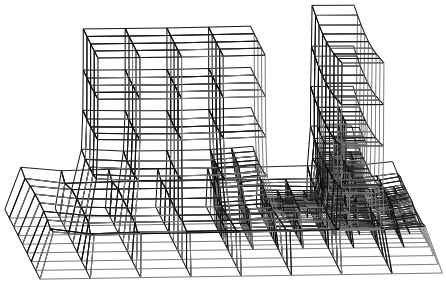
(b) Iteration 4



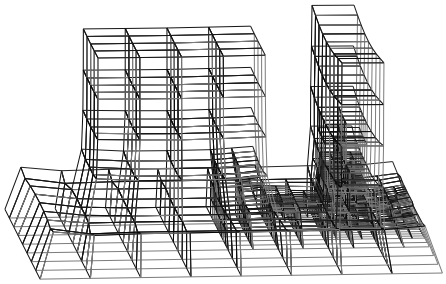
(c) Iteration 5



(d) Iteration 6

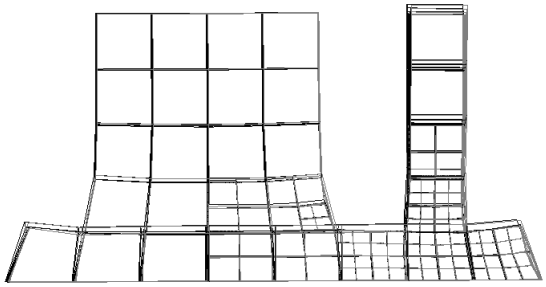


(e) Iteration 7

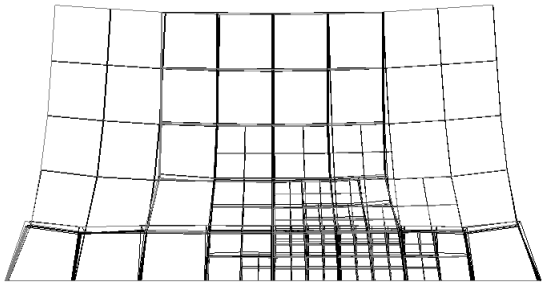


(f) Iteration 8

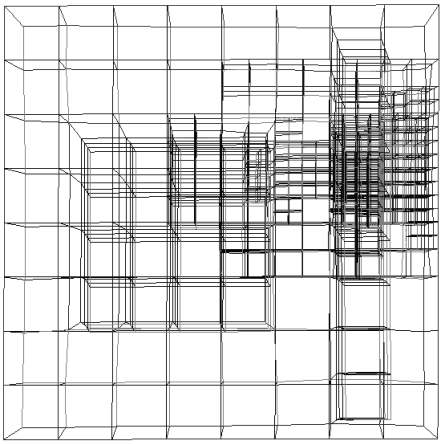
Figure 6.40: Goal-oriented mesh adaptivity for a 3-D lattice shown in Figure 6.23 with fixed base boundary condition. The goal is the distance between the two blocks measured near the bottom. Previous iterations are shown in Figure 6.39. Compare energy-oriented mesh adaptivity in Figures $6.34 - 6.35$.



(a) Front view

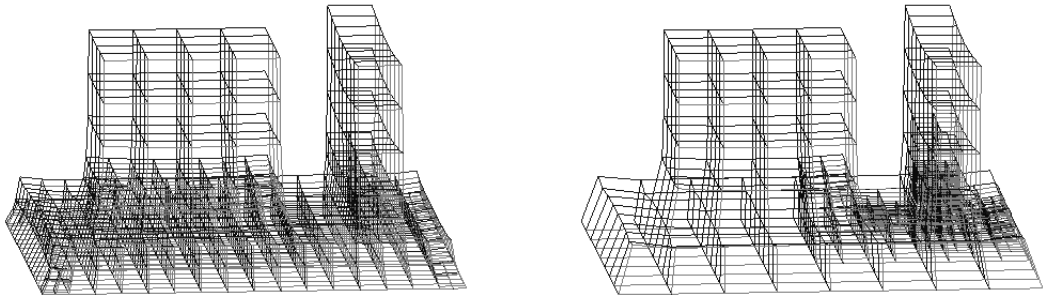


(b) Side view



(c) Top view

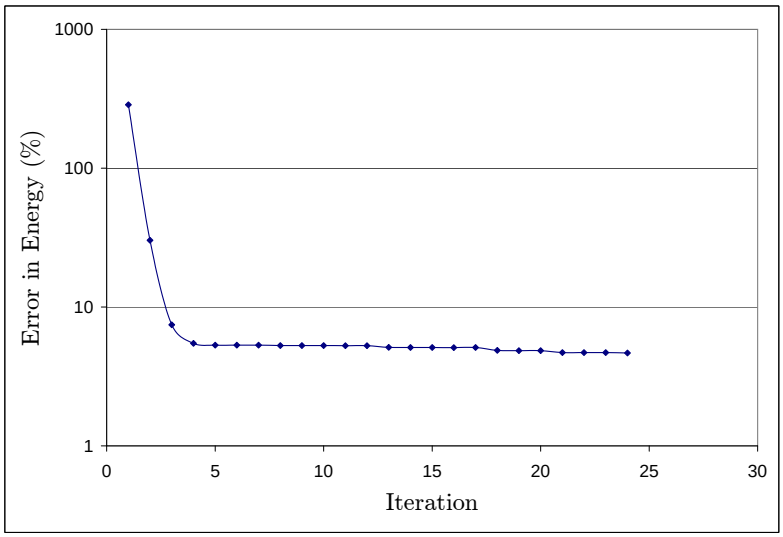
Figure 6.41: Front, side, and top views of converged mesh generated by goal-oriented adaptivity iterations shown in Figures 6.39 and 6.40. The goal is the distance between the two blocks measured near the bottom. Compare with Figure 6.36.



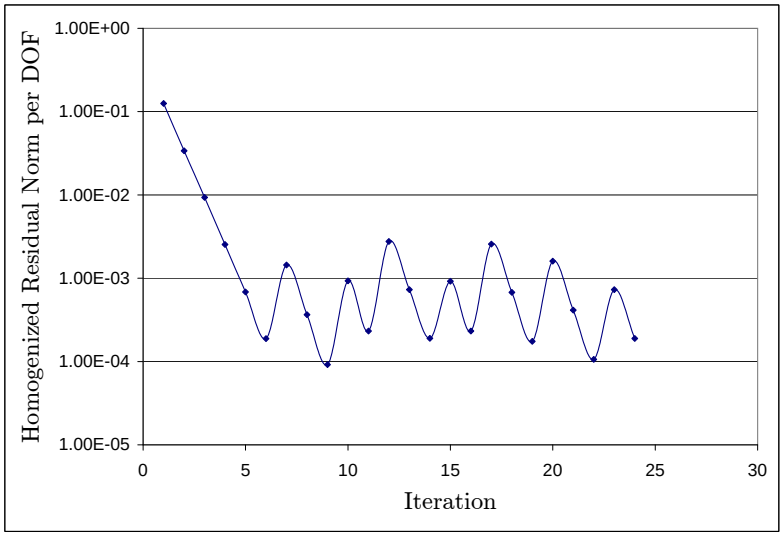
(a) Energy-oriented adaptivity (b) Goal-oriented adaptivity

Figure 6.42: Final meshes generated by energy-oriented adaptivity and goal-oriented adaptivity. The goal is the distance between the two blocks measured near the bottom.

Figure 6.42 shows the meshes for the converged solution for energy-oriented and goal-oriented adaptivity. Figure 6.43(a) shows convergence of error, measured as the difference of energy for current mesh and the minimum energy, as Newton iterations and mesh iterations proceed. The kinks are at some of the iterations where the mesh is refined because the solution has numerically converged for the current mesh. The curve is flat and has no kinks because the primary purpose of iterations is to reduce the error in goal and not reduce error in the energy. Compare the curve shown in Figure 6.37(a). In Figure 6.43(b), we plot the homogenized residual per DOF at each iteration. By homogenized residual we mean the vector that drives the Newton iterations. We divide by the number of DOFs because as the mesh is refined the number of DOFs increases. The residual goes up again upon mesh refinements because further details of the fine-scale are seen.



(a) Error in energy (%)

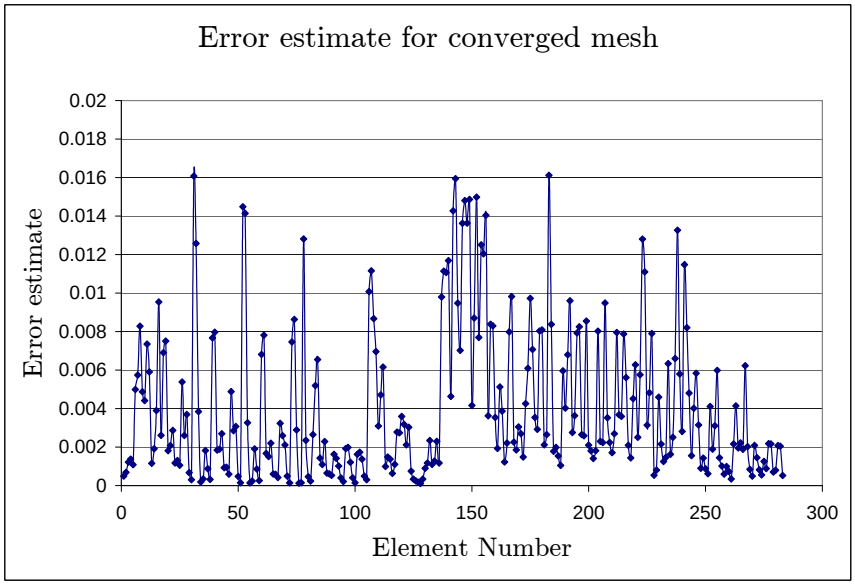


(b) Homogenized residual norm per DOF

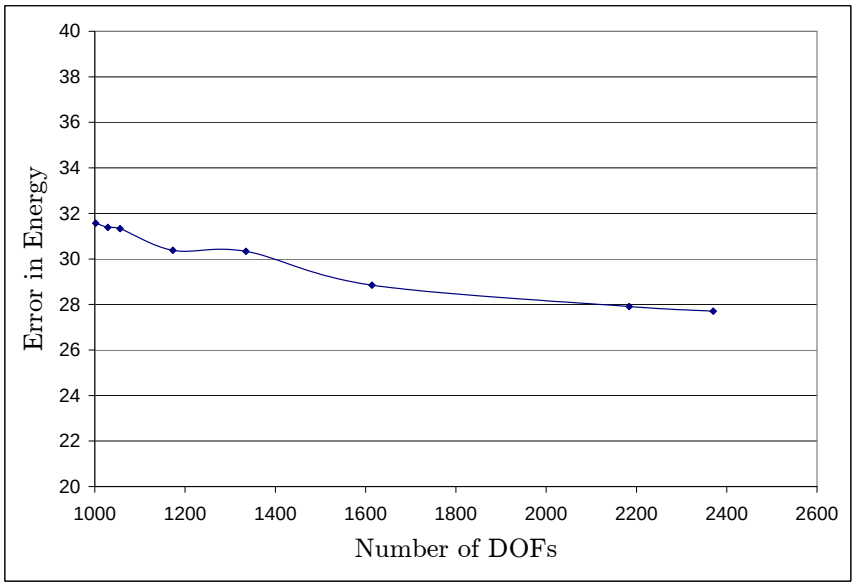
Figure 6.43: The graphs show how the minimum energy and residual norm change as mesh is refined using goal-oriented adaptivity and Newton iterations proceed. (a) The kinks are present where mesh is refined and the instantaneous rate of decrease of energy is higher. (b) The residual decreases with Newton iterations and goes up again upon mesh refinements. This is because fine-scale becomes more important. Compare with Figure 6.37.

Figure 6.44(a) shows *a posteriori* error estimates in various elements for the last mesh. The initial part of the curve is for some of the original mesh elements that have not been refined. Since the adjoint solution is small in some of them, they stay coarse. Compare Figure 6.38(a) for the distribution of the error when we measure it in energy-norm. Figure 6.44(b) is a plot of absolute error versus number of DOFs as the refinements proceed. The error is almost constant throughout the iterations. Compare this with the plot in Figure 6.44(b) for energy-oriented adaptivity in which the corresponding curve does not have flat regions.

The error estimate in goal, however, decreases uniformly as seen in Figure 6.45(a). We also plot the error estimates for individual elements in Figure 6.45(b). The estimate is divided into two parts − the standard characterization and the term due to homogenization (Section 4.10). The error due to homogenization is larger by around 2 orders of magnitude.

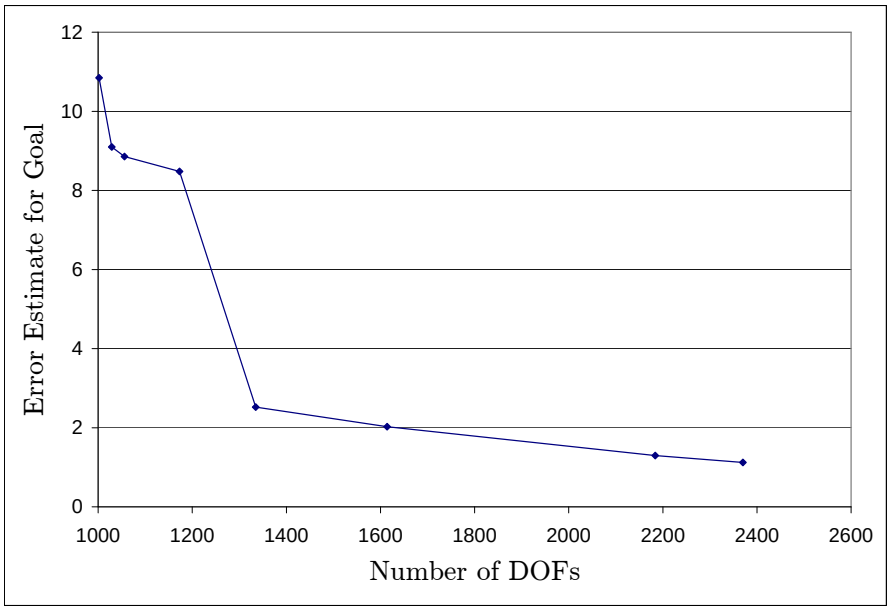


(a) A posteriori error estimates in various elements

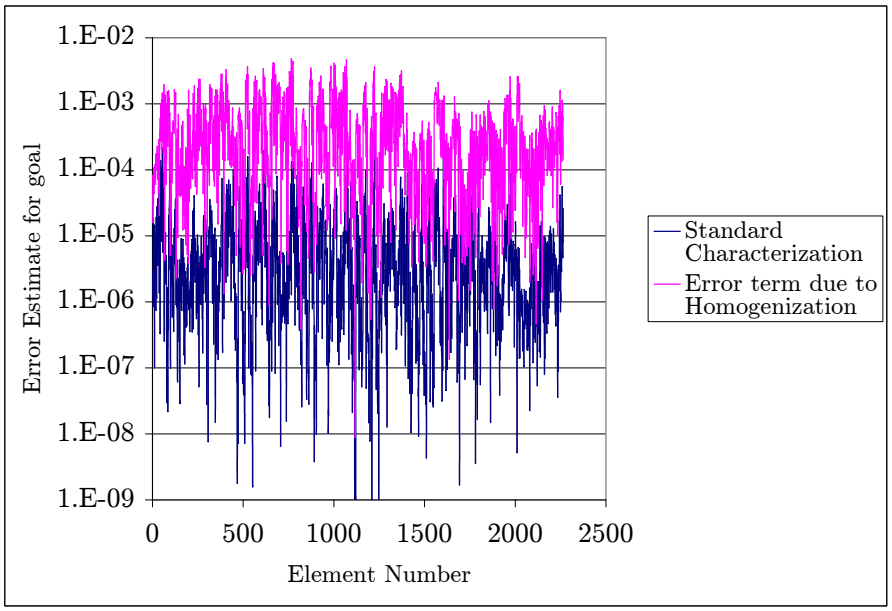


(b) Error in energy versus number of DOFs

Figure 6.44: The graphs show the distribution of errors in various elements and the decrease of error in energy with the increasing number of DOFs for goal-oriented adaptivity. Compare with Figure 6.38.



(a) Error estimate in goal versus number of DOFs



(b) Components of error estimate as discussed in Section 4.10

Figure 6.45: The decrease in error estimate for goal-oriented mesh adaptivity and components of error estimate as discussed in Section 4.10.

Chapter 7

Concluding Remarks and Directions for Future Research

The objective of this dissertation was to develop and analyze a black-box method for solving nonlinear problems with multiple scales due to rapidly varying material data. We have achieved this using the following key ideas – approximate mathematical models, error estimation with respect to the base model, adaptive meshes, a local numerical homogenization method, and fast algorithms for Moore-Penrose pseudoinverse.

Given a mesh that resolves the fine-scale, our method consists of creating a coarse mesh and numerically homogenizing the fine-scale material properties, that are linearized for a nonlinear problem, on each element separately. For each coarse-mesh element, the numerically homogenized material properties are computed by solving a local constrained convex optimization problem. The primary input is the Moore-Penrose pseudoinverse of local fine-scale stiffness matrix. The solution of this problem is a coarse-scale material compliance matrix. We refine the coarse-mesh based on an *a posteriori* error estimate until convergence.

We have shown that the optimization problem that defines the local

homogenization method is well-posed and has a unique solution. We have also shown that, for a certain choice of norm, homogenization preserves positive-definiteness of the fine-scale operator. We verified the method for a material with chess-board conductivity pattern. The possible choices for local homogenization were analyzed and it was seen that specific choices of local load and norm can give better results. For computational feasibility, it was required that we use fast algorithms for computing Moore-Penrose pseudoinverse of sparse singular finite element matrices. We presented various algorithms that were tailored for the purposes of this research. We studied the computational performance of the method for various methods of computing pseudoinverses and elements of various sizes. We derived optimal element sizes for homogenization in the context of SFIL. Finally, we integrated mesh adaptivity and nonlinear iterations with local homogenization. Convergence for both energy-oriented and goal-oriented adaptivity was demonstrated.

This research was carried out with a specific physical problem in mind, that of polymer shrinkage in Step and Flash Imprint Lithography process. Moreover, the local homogenization method was used in a very specific way − to create a coarse-scale Hessian matrix for finite element assembly. However, this work can be applied to a wide class of problems in a variety of ways and leads to many ideas for future research.

- As long as a suitable interpolation operator is used, it can be applied to different kinds of finite elements. With further research, the method

developed here can be put to various other uses. For example, the coarse-scale stiffness matrix can be used simply as a preconditioner for solving the base model. Local homogenization can also be used as an indicator to check whether locally the mesh is sufficiently refined (without solving a global problem).

- Currently, the local constrained convex optimization problems are solved approximately using a regularization approach using the parameter ϵ. Although this has worked well and is justified due to the iterative nature of the algorithm, it is not difficult to formulate the exact problem. It is a linear system for constrained optimization with blocks consisting of Kronecker product matrices. The problem how is to solve this efficiently and look for any advantages it can provide. For example, each step of the Newton iterations may be better in reducing the energy.

- We have fixed the choice of the local interpolation operator and norm for measuring the local error using physical intuition. The accuracy of these choices has been verified using numerical experiments. However, it will be better if the intuition can be mathematically justified.

- Further analysis needs to be done to prove spectral properties of the homogenized stiffness matrix when the error is minimized in different norms or whether the specific load formulation of the error is used. We have proved that the homogenized stiffness matrix is positive semi-definite

when the fine-scale stiffness matrix is positive semi-definite and the error is minimized in the ℓ_2 norm and all loads are treated equally.

- Although we have tried three different algorithms for Moore-Penrose pseudoinverse, we have not integrated the fourth one which uses "proper splittings" to compute pseudoinverse of a modified matrix. It would require more memory but it might make the overall method faster.

- We have implemented the algorithm for certain finite elements, specifically for quads and hexes with polynomial degree 1. To be widely applicable, we need to test and implement it for other elements and with higher polynomial degrees.

- For larger systems, the homogenization can be done in parallel on multiple processors. We have done preliminary research to empirically determine the optimal time to homogenize elements of various sizes. This will be useful in allocating elements to different processors so that the load is shared equally.

Appendices

Appendix A

Local Numerical Homogenization – The Stationary Point of the Lagrangian

We derive conditions for stationarity of the Lagrangian, which is defined in Equation (4.5), for the local homogenization problem (Section 4.3.2).

For $K^\dagger \in \mathbb{R}^{N\times N}$, $K^\dagger$ symmetric, $A \in \mathbb{R}^{N\times M}$, $X \in \mathbb{R}^{M\times M}$, $B \in \mathbb{S}^N_{++}$, $f \in \mathbb{R}^N - \{0\}$, $\Lambda \in \mathbb{R}^{M\times M}$, and $\epsilon > 0$ the Lagrangian is

$$\mathcal{L}(X,\Lambda) = \frac{1}{2}\left|\left|(K^\dagger - AXA^T)f\right|\right|_B^2 \; + \; \frac{\epsilon}{2}\left|\left|K^\dagger - AXA^T\right|\right|_{F,B}^2 ||f||_2^2$$

$$+\text{trace}(\Lambda^T(X - X^T))$$

$$\begin{aligned}
&= \frac{1}{2}\, f^T K^\dagger B K^\dagger f + \frac{\epsilon}{2}\,||f||_2^2 \,\text{trace}\left(K^\dagger B K^\dagger\right) && \Big\}1\\
&+ \text{trace}(\Lambda^T(X - X^T)) - f^T K^\dagger B A X A^T f - \epsilon\,||f||_2^2 \,\text{trace}\left(K^\dagger B A X A^T\right) && \Big\}2\\
&+ \frac{1}{2} f^T A X^T (A^T B A) X A^T f + \frac{\epsilon}{2}\,||f||_2^2 \,\text{trace}\left(A X^T (A^T B A) X A^T\right). && \Big\}3
\end{aligned}$$

Here 1, 2, and 3 denote the constant, linear, and quadratic terms respectively.

A.1 Derivatives of the linear terms

We differentiate $\mathcal{L}(X, \Lambda)$ with respect to Λ_{ij}. δ denotes the Kronecker delta symbol.

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \Lambda_{ij}} &= \frac{\partial}{\partial \Lambda_{ij}} \left(\text{trace}(\Lambda^T (X - X^T)) \right) \\
&= \frac{\partial}{\partial \Lambda_{ij}} \left(\sum_{k,l} \Lambda_{lk} (X_{lk} - X_{kl}) \right) \\
&= \sum_{k,l} \delta_{il} \delta_{jk} (X_{lk} - X_{kl}) \\
&= X_{ij} - X_{ji}
\end{aligned}$$

Thus, $\frac{\partial \mathcal{L}}{\partial \Lambda} = X - X^T$ and we recover the symmetry constraint on X.

We now differentiate the linear terms of $\mathcal{L}(X, \Lambda)$ with respect to X_{ij}.

$$\begin{aligned}
\frac{\partial}{\partial X_{ij}} \left(\text{trace}(\Lambda^T (X - X^T)) \right) &= \frac{\partial}{\partial X_{ij}} \left(\sum_{k,l} \Lambda_{lk} (X_{lk} - X_{kl}) \right) \\
&= \sum_{k,l} \Lambda_{lk} (\delta_{li} \delta_{kj} - \delta_{ki} \delta_{lj}) \\
&= \Lambda_{ij} - \Lambda_{ji}
\end{aligned}$$

Thus, $\frac{\partial}{\partial X} \left(\text{trace}(\Lambda^T (X - X^T)) \right) = \Lambda - \Lambda^T$.

For $u, v \in \mathbb{R}^M$, we have

$$\begin{aligned}
\frac{\partial}{\partial X_{ij}} \left(u^T X v \right) &= \frac{\partial}{\partial X_{ij}} \left(\sum_{k,l} X_{kl} v_l u_k \right) \\
&= \sum_{k,l} \delta_{ki} \delta_{lj} v_l u_k \\
&= u_i v_j .
\end{aligned}$$

Thus, $\frac{\partial}{\partial X} \left(u^T X v \right) = u v^T$.

For $C \in \mathbb{R}^{N\times M}$ and $D \in \mathbb{R}^{M\times N}$, we have

$$\begin{aligned}
\frac{\partial}{\partial X_{ij}}\left(\text{trace}(CXD)\right) &= \frac{\partial}{\partial X_{ij}}\left(\sum_{k,l,m} C_{kl}X_{lm}D_{mk}\right) \\
&= \sum_{k,l,m} C_{kl}\delta_{li}\delta_{mj}D_{mk} \\
&= \sum_{k} C_{ki}D_{jk} \\
&= (DC)_{ji} \\
&= (DC)^T_{ij}.
\end{aligned}$$

Thus, $\dfrac{\partial}{\partial X}\left(\text{trace}(CXD)\right) = (DC)^T = C^T D^T$.

A.2 Derivatives of the quadratic terms

We now differentiate the quadratic terms of $\mathcal{L}(X,\Lambda)$ with respect to X_{ij}.

For $v \in \mathbb{R}^M$ and $F \in \mathbb{R}^{M\times M}$, we have

$$\begin{aligned}
\frac{\partial}{\partial X_{ij}}\left(v^T X^T F X v\right) &= \frac{\partial}{\partial X_{ij}}\left(\sum_{k,l,m,n} F_{kl}v_m v_n X_{lm}X_{kn}\right) \\
&= \sum_{k,l,m,n} F_{kl}v_m v_n\left(\delta_{il}\delta_{jm}X_{kn} + \delta_{ki}\delta_{jn}X_{lm}\right) \\
&= 2\sum_{k,n} F_{ki}v_j v_n X_{kn} \\
&= 2(FXv)_i v_j.
\end{aligned}$$

Thus, $\dfrac{\partial}{\partial X}\left(v^T X^T F X v\right) = 2FXvv^T$.

For $G \in \mathbb{R}^{N\times M}$ and $H \in \mathbb{R}^{M\times M}$, we have

$$\begin{aligned}
\frac{\partial}{\partial X_{ij}}\left(\text{trace}(GX^THXG^T)\right) &= \frac{\partial}{\partial X_{ij}}\left(\sum_{k,l,m,n,p} G_{kl}G_{kp}H_{mn}X_{ml}X_{np}\right) \\
&= \sum_{k,l,m,n,p} G_{kl}G_{kp}H_{mn}(\delta_{mi}\delta_{lj}X_{np} + \delta_{ni}\delta_{pj}X_{ml}) \\
&= 2\sum_{k,n,p} G_{kj}G_{kp}H_{in}X_{np} \\
&= 2(HXG^TG)_{ij}.
\end{aligned}$$

Thus, $\frac{\partial}{\partial X}\left(\text{trace}(GX^THXG^T)\right) = 2HXG^TG$.

A.3 The stationarity conditions

Choosing

$$\begin{aligned}
u &= A^TBK^\dagger f \\
v &= A^Tf \\
C &= K^\dagger BA \\
D &= A^T \\
F &= A^TBA \\
G &= A \\
H &= A^TBA
\end{aligned}$$

in the equations above, we derive the stationarity conditions for the Lagrangian. We get $X = X^T$ and Equation (4.6), which is reproduced here.

$$(A^TBA)X(A^T(ff^T + \epsilon\,||f||_2^2\,I)A) - A^TBK^\dagger(ff^T + \epsilon\,||f||_2^2\,I)A = \Lambda^T - \Lambda$$

Appendix B

Continuum Approximation of Nonlinear Lattice Elasticity

We derive an approximate continuum hyperelasticity model for nonlinear elasticity of a lattice with particles interacting by central pair-potentials with neighbors. As the number of lattice points increases in each spatial direction, the lattice is approximated as a continuum.

B.1 Motivation

Many multiscale problems lack analytical macroscopic or coarse-scale models. Individually tailored macroscopic models have to be derived from the underlying precise but expensive fine-scale models. The derived coarse-scale models can be used together with the fine-scale model by using many coupling approaches [40, 81, 16, 21]. Model adaptivity can be used to choose appropriate models, coarse or fine, in different spatial or temporal regions to control the modeling error [69].

B.2 Derivation of a hyperelasticity model from the lattice model

For simplicity, we will work with harmonic potentials only. Other potentials can be derived in a similar manner.

Consider a cuboidal lattice with n cells in each direction (Figure B.1). Let $\mathcal{A}$ be the set of all particles. $\mathcal{A} = \mathcal{A}^I \cup \mathcal{A}^S$ where $\mathcal{A}^I$ is the set of interior particles and $\mathcal{A}^S$ is the set of surface particles. $\mathcal{A}^I \cap \mathcal{A}^S = \phi$. We have $\left|\mathcal{A}^I\right| = O(n^3)$ and $\left|\mathcal{A}^S\right| = O(n^2)$.

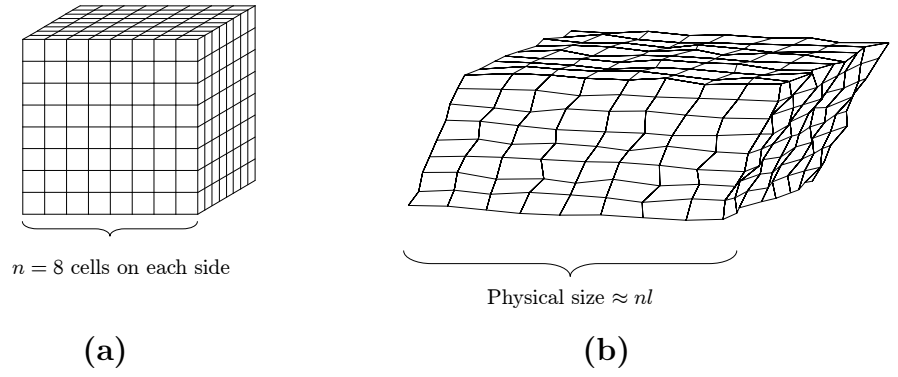


Figure B.1: A cuboidal lattice and (a) its enumeration/reference domain and (b) its physical domain for a particular deformation. Only "edge" bonds are shown. l is a characteristic bond length.

Let $\mathbf{x}_\alpha, \alpha \in \mathcal{A}$ denote the unknown position vector of particle α. Let $k_{\alpha\beta}$ and $l_{\alpha\beta}$ be the stiffness and equilibrium length respectively of the spring connecting α and β. Let $\mathbf{F}_\alpha^{EXT}$ stand for the external force on particle α. The

total potential energy of the lattice is

$$\left.\begin{aligned} J\left(\{\mathbf{x}_\alpha\}_{\alpha\in\mathcal{A}}\right) &= \frac{1}{2}\sum_{\alpha\in\mathcal{A}}\sum_{\substack{\beta\in\mathcal{A}\\ \alpha\leftrightsquigarrow\beta}}\frac{k_{\alpha\beta}}{2}\left(||\mathbf{x}_\alpha-\mathbf{x}_\beta||-l_{\alpha\beta}\right)^2 \\ &- \sum_{\alpha\in\mathcal{A}^I}\mathbf{F}_\alpha^{EXT,\,I}\cdot\mathbf{x}_\alpha \\ &- \sum_{\alpha\in\mathcal{A}^S}\mathbf{F}_\alpha^{EXT,\,S}\cdot\mathbf{x}_\alpha. \end{aligned}\right\}\quad \begin{aligned} &O(n^3)\text{ terms}\\ &O(n^3)\text{ terms}\\ &O(n^2)\text{ terms}\end{aligned}$$

Hence, for large n, the total potential energy is $O(n^3kl^2)+O(n^3F^Il)+O(n^2F^Sl)$ where l is a characteristic spring length, k a characteristic spring constant, F^I a characteristic interior force, and F^S a characteristic surface force. As n increases, the physical dimensions of the lattice in each direction grow as $O(nl)$.

We want to approximate the hyperelastic behavior of the lattice as $n\to\infty$. But in this limit the total potential energy and other quantities are infinite. To make further analysis possible, we scale the physical quantities appropriately to derive a model with finite energy.

B.2.1 Dimensional scaling of the lattice

We have to scale the parameters appropriately in the total potential energy function to keep the stored energy finite and keep the magnitude of stored energy in balance with the external work done by the forces. We use a hat to mark scaled quantities. The following scaling is the one that achieves

these constraints.

$$\begin{aligned}
\mathbf{x}_\alpha/n &= \widehat{\mathbf{x}}_\alpha \\
l_{\alpha\beta}/n &= \widehat{l}_{\alpha\beta} \\
k_{\alpha\beta}/n &= \widehat{k}_{\alpha\beta} \\
\mathbf{F}_\alpha^{EXT,\,I}/n^2 &= \widehat{\mathbf{F}}_\alpha^{EXT,\,I} \\
\mathbf{F}_\alpha^{EXT,\,S}/n^2 &= \widehat{\mathbf{F}}_\alpha^{EXT,\,S} \\
J/n^3 &= \widehat{J}
\end{aligned}$$

These modifications change the expression for total potential energy to

$$\begin{aligned}
\widehat{J}\left(\{\widehat{\mathbf{x}}_\alpha\}_{\alpha\in\mathcal{A}}\right) &= \frac{1}{2}\sum_{\alpha\in\mathcal{A}}\sum_{\substack{\beta\in\mathcal{A}\\ \alpha\leftrightsquigarrow\beta}} \underbrace{\frac{\widehat{k}_{\alpha\beta}}{2}\left(||\widehat{\mathbf{x}}_\alpha-\widehat{\mathbf{x}}_\beta||-\widehat{l}_{\alpha\beta}\right)^2}_{O(1/n^3)\text{ each}} \Bigg\} \quad O(n^3)\text{ terms} \\
&- \sum_{\alpha\in\mathcal{A}^I} \underbrace{\widehat{\mathbf{F}}_\alpha^{EXT,\,I}\cdot\widehat{\mathbf{x}}_\alpha}_{O(1/n^3)\text{ each}} \Bigg\} \quad O(n^3)\text{ terms} \\
&- \sum_{\alpha\in\mathcal{A}^S} \underbrace{\widehat{\mathbf{F}}_\alpha^{EXT,\,S}\cdot\widehat{\mathbf{x}}_\alpha}_{O(1/n^3)\text{ each}}. \Bigg\} \quad O(n^2)\text{ terms}
\end{aligned}$$

After applying the scaling, the physical dimensions of the lattice in each direction become $O(l)$ and the order of the scaled total potential energy $\widehat{J}$ is independent of n. Hence, as $n \to \infty$, we will take limits of finite quantities. Moreover, the summations in the expression for $\widehat{J}$ look like Riemann sums for volume and surface integrals.

B.2.2 Continuum stored energy density and external work

We derive integrals as formal limits of the summations present in the expression for scaled total potential energy $\widehat{J}$. For simplicity and in anticipation of hyperelasticity equations, the notation is changed slightly.

In the unscaled lattice with n cells on each side (and hence $n+1$ particles on each side), we identify each particle by a triple of integers (i_1, i_2, i_3) where $i_j \in [0, n], j = 1, 2, 3$. For the scaled lattice, the particles are enumerated by scaled real numbers. The scaling in enumeration domain maps particle (i_1, i_2, i_3) to $\boldsymbol{\xi} := (l\, i_1/n, l\, i_2/n, l\, i_3/n)$. Hence, apart from the physical lattice remaining bounded as $n \to \infty$, the reference domain is also bounded (Figure B.1). We define the set of all particles

$$\Omega = \left\{ \boldsymbol{\xi} : \boldsymbol{\xi} = (\frac{l\, i_1}{n}, \frac{l\, i_2}{n}, \frac{l\, i_3}{n}) \text{ where } 0 \leq i_j \leq n \text{ for } j = 1, 2, 3 \right\}.$$

Ω^I and Ω^S represent interior and surface particles respectively.

Let $\boldsymbol{x}(\boldsymbol{\xi})$ be the position vector of particle $\boldsymbol{\xi}$ in a particular physical configuration of the lattice; $\boldsymbol{x} : \mathbb{R}^3 \to \mathbb{R}^3$. Let $\mathcal{X} := \{\boldsymbol{x}(\boldsymbol{\xi}) : \boldsymbol{\xi} \in \Omega\}$ be the set of physical coordinates.

Let $E_{\boldsymbol{\xi}\boldsymbol{\xi}'}$ be the central potential function between particles $\boldsymbol{\xi}$ and $\boldsymbol{\xi}'$. For harmonic springs,

$$E_{\boldsymbol{\xi}\boldsymbol{\xi}'}\left(\boldsymbol{x}(\boldsymbol{\xi}), \boldsymbol{x}(\boldsymbol{\xi}')\right) = \frac{1}{2}\widehat{k}_{\boldsymbol{\xi}\boldsymbol{\xi}'}\left(||\boldsymbol{x}(\boldsymbol{\xi}') - \boldsymbol{x}(\boldsymbol{\xi})|| - \widehat{l}^0_{\boldsymbol{\xi}\boldsymbol{\xi}'}\right)^2.$$

We can now express the total potential energy (TPE) as a function of $\mathcal{X}$.

$$\mathrm{TPE}(\mathcal{X}) = \frac{1}{2}\sum_{\boldsymbol{\xi}\in\Omega}\sum_{\substack{\boldsymbol{\xi}'\in\Omega \\ \boldsymbol{\xi}\rightsquigarrow\boldsymbol{\xi}'}} E_{\boldsymbol{\xi}\boldsymbol{\xi}'}\left(\boldsymbol{x}(\boldsymbol{\xi}), \boldsymbol{x}(\boldsymbol{\xi}')\right) - \sum_{\boldsymbol{\xi}\in\Omega^I} \widehat{\boldsymbol{F}}(\boldsymbol{\xi})\cdot\boldsymbol{x}(\boldsymbol{\xi}) - \sum_{\boldsymbol{\xi}\in\Omega^S} \widehat{\boldsymbol{F}}(\boldsymbol{\xi})\cdot\boldsymbol{x}(\boldsymbol{\xi}), \tag{B.1}$$

where $\widehat{\boldsymbol{F}}(\boldsymbol{\xi})$ is the force on particle $\boldsymbol{\xi}$. For a lattice with nearest neighbors, we have

$$\boldsymbol{\xi}' = \boldsymbol{\xi} + \left(\frac{l\,p_1}{n}, \frac{l\,p_2}{n}, \frac{l\,p_3}{n}\right)$$

where $\boldsymbol{p} \in \mathbb{R}^3$ represents the direction of the neighbor in the enumeration domain (Figure 2.5). $p_j \in \{-1, 0, 1\}$ for $j = 1, 2, 3$.

$$\sum_{j=1}^{3} |p_j| = \begin{cases} 1 & \text{for edges} \\ 2 & \text{for face diagonals} \\ 3 & \text{for cube diagonals} \end{cases}$$

As $n \to \infty$, $||\boldsymbol{\xi}' - \boldsymbol{\xi}|| \to 0$. We formally extend $\boldsymbol{x}$ defined on Ω, which is a discrete subset of $[0, l]^3$, to the full cube $[0, l]^3$ as $n \to \infty$. We assume it is twice continuously differentiable with respect to $\boldsymbol{\xi} \in [0, l]^3$ and use the Taylor series expansion of $\boldsymbol{x}$ in $\boldsymbol{\xi}$.

$$\left|\left|\boldsymbol{x}(\boldsymbol{\xi}) - \boldsymbol{x}(\boldsymbol{\xi}') - \frac{l}{n}\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}}\boldsymbol{p}\right|\right| \sim \frac{1}{n^2}\left|\left|\frac{\partial^2 \boldsymbol{x}}{\partial \boldsymbol{\xi}^2}\right|\right| \tag{B.2}$$

Here $\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}}$ is the Jacobian (the deformation gradient). $\frac{\partial^2 \boldsymbol{x}}{\partial \boldsymbol{\xi}^2}$ represents the second order derivatives. Ignoring the higher order terms and using this approximation in the potential energy expression for the harmonic springs, we get

$$E_{\boldsymbol{\xi}\boldsymbol{\xi}'}\left(\boldsymbol{x}(\boldsymbol{\xi}), \boldsymbol{x}(\boldsymbol{\xi}')\right) \approx \frac{k_{\boldsymbol{\xi}\boldsymbol{\xi}'}}{2n^3}\left(l\left|\left|\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}}\boldsymbol{p}\right|\right| - l^0_{\boldsymbol{\xi}\boldsymbol{\xi}'}\right)^2. \tag{B.3}$$

Expanding the norm inside, we get $\left|\left|\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}}\boldsymbol{p}\right|\right| = \sqrt{\boldsymbol{p}^T C \boldsymbol{p}}$ where

$$C = \left(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}}\right)^T \left(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}}\right) = \begin{pmatrix} \partial \boldsymbol{x}/\partial\xi_1 \cdot \partial \boldsymbol{x}/\partial\xi_1 & \partial \boldsymbol{x}/\partial\xi_1 \cdot \partial \boldsymbol{x}/\partial\xi_2 & \partial \boldsymbol{x}/\partial\xi_1 \cdot \partial \boldsymbol{x}/\partial\xi_3 \\ \partial \boldsymbol{x}/\partial\xi_2 \cdot \partial \boldsymbol{x}/\partial\xi_1 & \partial \boldsymbol{x}/\partial\xi_2 \cdot \partial \boldsymbol{x}/\partial\xi_2 & \partial \boldsymbol{x}/\partial\xi_2 \cdot \partial \boldsymbol{x}/\partial\xi_3 \\ \partial \boldsymbol{x}/\partial\xi_3 \cdot \partial \boldsymbol{x}/\partial\xi_1 & \partial \boldsymbol{x}/\partial\xi_3 \cdot \partial \boldsymbol{x}/\partial\xi_2 & \partial \boldsymbol{x}/\partial\xi_3 \cdot \partial \boldsymbol{x}/\partial\xi_3 \end{pmatrix}.$$

$C(\boldsymbol{\xi})$ is the right Cauchy-Green deformation tensor at $\boldsymbol{\xi}$ in the reference domain and $\boldsymbol{x}$ is interpreted as motion.

Looking at the expression for potential energy for a single bond in Equation (B.3), we see n^3 in the denominator. Since we have $O(n^3)$ bonds, and each particle is connected to nearest neighbors only, the stored energy summation can be converted to a volume integral. We have to split the summations into different parts with each corresponding to a different neighbor type. As we do not have cube-diagonal neighbors in our model, we ignore that part (Figure 2.5).

$$\sum_{\boldsymbol{\xi}\in\Omega} \sum_{\substack{\boldsymbol{\xi}'\in\Omega \\ \boldsymbol{\xi}\rightsquigarrow\boldsymbol{\xi}'}} \equiv \sum_{\boldsymbol{\xi}\in\Omega} \sum_{\substack{\text{6 edge} \\ \text{bonds}}} + \sum_{\boldsymbol{\xi}\in\Omega} \sum_{\substack{\text{12 face} \\ \text{bonds}}} + \sum_{\boldsymbol{\xi}\in\Omega} \sum_{\substack{\text{8 cube} \\ \text{bonds}}} 0$$

The sum for edge bonds with representative spring constant k_E and representative equilibrium length l_E^0 is

$$\frac{1}{2} \sum_{\boldsymbol{\xi}\in\Omega} \sum_{\substack{\text{6 edge} \\ \text{bonds}}} E_{\boldsymbol{\xi}\boldsymbol{\xi}'}\left(\boldsymbol{x}(\boldsymbol{\xi}), \boldsymbol{x}(\boldsymbol{\xi}')\right) = \frac{1}{2n^3} \sum_{\boldsymbol{\xi}\in\Omega} \sum_{j=1}^{3} k_E \left(l\sqrt{C_{jj}} - l_E^0\right)^2.$$

The sum for face diagonal bonds with representative spring constant k_F and

representative equilibrium length l_F^0 is

$$\frac{1}{2}\sum_{\boldsymbol{\xi}\in\Omega}\sum_{\substack{12\text{ face}\\ \text{bonds}}} E_{\boldsymbol{\xi}\boldsymbol{\xi}'}\left(\boldsymbol{x}(\boldsymbol{\xi}),\boldsymbol{x}(\boldsymbol{\xi}')\right) = \frac{1}{2n^3}\sum_{\boldsymbol{\xi}\in\Omega} k_F \Bigg[$$
$$\left(l\sqrt{C_{11}+C_{22}+2C_{12}}-l_F^0\right)^2+\left(l\sqrt{C_{11}+C_{22}-2C_{12}}-l_F^0\right)^2$$
$$+\left(l\sqrt{C_{22}+C_{33}+2C_{23}}-l_F^0\right)^2+\left(l\sqrt{C_{22}+C_{33}-2C_{23}}-l_F^0\right)^2$$
$$+\left(l\sqrt{C_{33}+C_{11}+2C_{31}}-l_F^0\right)^2+\left(l\sqrt{C_{33}+C_{11}-2C_{31}}-l_F^0\right)^2 \Bigg].$$

If we approximate the summations with integrals as for large n and ignore $o(1/n)$ terms, we get

$$\mathrm{TPE}(\boldsymbol{x}) = \int_{V_0} W_L(C)\,\mathrm{d}V_0 - \int_{V_0} f^{EXT}\cdot\boldsymbol{x}\,\mathrm{d}V_0 - \int_{S_0} t^{EXT}\cdot\boldsymbol{x}\,\mathrm{d}S_0$$

where V_0 is $[0,l]^3$, S_0 is its surface, and the stored energy density in terms of the right Cauchy-Green deformation tensor is

$$\begin{aligned}
W_L(C) = \tfrac{1}{2}[\; &+k_E(l\sqrt{C_{11}}-l_E^0)^2 && (\pm1,0,0) \\
&+k_E(l\sqrt{C_{22}}-l_E^0)^2 && (0,\pm1,0) \quad \text{Direction set 1, } ||p||_1=1 \\
&+k_E(l\sqrt{C_{33}}-l_E^0)^2 && (0,0,\pm1) \\
&+k_F(l\sqrt{C_{11}+C_{22}-2C_{12}}-l_F^0)^2 && \\
&+k_F(l\sqrt{C_{11}+C_{22}+2C_{12}}-l_F^0)^2 && (\pm1,\pm1,0) \\
&+k_F(l\sqrt{C_{22}+C_{33}-2C_{23}}-l_F^0)^2 && \\
&+k_F(l\sqrt{C_{22}+C_{33}+2C_{23}}-l_F^0)^2 && (0,\pm1,\pm1) \quad \text{Direction set 2, } ||p||_1=2 \\
&+k_F(l\sqrt{C_{33}+C_{11}-2C_{31}}-l_F^0)^2 && \\
&+k_F(l\sqrt{C_{33}+C_{11}+2C_{31}}-l_F^0)^2] && (\pm1,0,\pm1)
\end{aligned}$$

Since the lattice is stochastic and some edge bonds can be non-covalent, there is no single k_E or l_E that would be valid everywhere. Hence, for the case

of interest, we still would have to determine the effective parameters in the strain energy density. The purpose of this analysis was to provide the functional form of the energy density that arises naturally from the lattice topology and bond potentials. The expression for energy density can be expanded in terms of unknown constants $\{a_i\}_{i=1}^4$.

$$\begin{aligned} W_L(C) &= a_1(C_{11}+C_{22}+C_{33}) \\ &+ a_2(\sqrt{C_{11}}+\sqrt{C_{22}}+\sqrt{C_{33}}) \\ &+ a_3(\sqrt{C_{11}+C_{22}-2C_{12}}+\sqrt{C_{11}+C_{22}+2C_{12}}) \\ &+ a_3(\sqrt{C_{22}+C_{33}-2C_{23}}+\sqrt{C_{22}+C_{33}+2C_{23}}) \\ &+ a_3(\sqrt{C_{33}+C_{11}-2C_{31}}+\sqrt{C_{33}+C_{11}+2C_{31}}) \\ &+ a_4 \end{aligned}$$

For a given discrete lattice, $\{a_i\}_{i=1}^4$ will be determined through the inverse analysis, by applying different boundary conditions and comparing energies of the lattice and the continuum.

B.3 Numerical results

Although we derived the functional form of the stored energy density for the continuum approximation of the lattice, we still have to compute the unknown coefficients $\{a_i\}_{i=1}^4$.

To this end, we choose a reasonably large cubical lattice and apply affine deformations to its boundary points (Figure B.2). We let the lattice relax to equilibrium and compute the total stored energy. Same deformation

is applied to the boundary of a cube that represents the continuum. We do know the resulting deformation inside the cube and can compute the stored energy for the continuum in terms of the coefficients $\{a_i\}_{i=1}^4$. Using linear least-squares matching of energies, the best coefficients can be obtained.

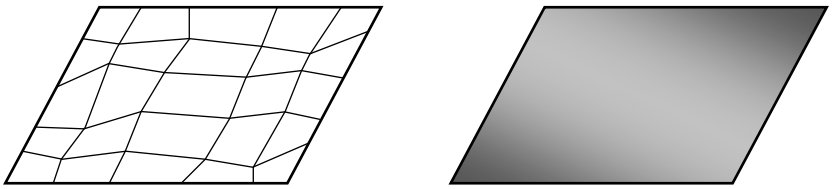


Figure B.2: (a) Lattice in equilibrium with boundary points fixed by an affine deformation and (b) Continuum with constant deformation tensor equal to average lattice deformation tensor.

We choose a lattice with 20 cells on each side and with spring parameters $k_E = 2, l_E^0 = 0.8, k_F = 0.5, l_F^0 = 0.8\sqrt{2}$. The components of the deformation gradient F range from 1 to 2 for diagonal entries and 0 to 0.5 for xy and xz off-diagonal entries in four equal increments.

$$F = \begin{bmatrix} 1:2 & 0:0.5 & 0:0.5 \\ 0 & 1:2 & 0 \\ 0 & 0 & 1:2 \end{bmatrix}$$

This gives total 3125 data points to compute the best fit. The best $\{a_i\}_{i=1}^4$ are determined to be $\{10088, -6188, -3838, 20810\}$ and the average misfit error between lattice and continuum energies for these best coefficients is 2.1%, indicating an acceptable match.

B.4 Conclusions

We can derive a continuum hyperelasticity model for approximating a lattice with nearest neighbors interacting via central potentials. This provides a natural continuum model for large scale lattice features. The analytically obtained form of the energy density depends on constants that can be found by numerical solution of the fine-scale discrete lattice on a reasonably large lattice.

Appendix C

On Skew-symmetry of Outer Product of Two Vectors

We prove that if the outer product of two vectors is skew-symmetric, then at least one of the two vectors is zero. We do not require the outer product to be identically zero to deduce that at least one of the vectors is zero. We only require a weaker condition (that the outer product be skew-symmetric).

Let $u, v \in \mathbb{R}^N$. Let uv^T be skew-symmetric. Thus $uv^T + vu^T = 0$. We show that either $u = 0$, or $v = 0$, or both can be zero, which is the trivial case. Without loss of generality, assume $u \neq 0$. We have the following implications.

$$\begin{aligned} & uv^T + vu^T && = && 0 \\ \Rightarrow \quad & (uv^T + vu^T)u && = && 0 \\ \Rightarrow \quad & u(v^T u) + v\,||u||_2^2 && = && 0 \\ \Rightarrow \quad & u(u^T v) + ||u||_2^2\, v && = && 0 \\ \Rightarrow \quad & (uu^T + ||u||_2^2\, I)v && = && 0 \end{aligned}$$

Now $||u||_2^2\, I$ is symmetric positive definite and uu^T is symmetric positive semi-definite. Thus, their sum is symmetric positive definite and its action on a vector can be zero iff the vector is 0. Hence $v = 0$ and we're done.

A different (and more elementary) way to prove this result is to explicitly write the elements of $uv^T + vu^T$ (in terms of u_i and v_i) and equate each of

them to 0 to deduce progressively that either all $u_i = 0$, or all $v_i = 0$, or both vectors are 0. We skip the details.

Bibliography

[1] M. Ainsworth and J. T. Oden. *A Posteriori Error Estimation in Finite Element Analysis.* Wiley-Interscience, 2000.

[2] G. Allaire. *Shape Optimization by the Homogenization Method.* Springer, 2002.

[3] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide.* SIAM, Philadelphia, PA, third edition, 1999.

[4] A. C. Antoulas. *Approximation of Large-Scale Dynamical Systems.* Cambridge University Press, 2005.

[5] T. Arbogast. Numerical subgrid upscaling of two-phase flow in porous media. In *Numerical Treatment of Multiphase Flows in Porous Media*, Lecture Notes in Physics, pages 35–49. Springer, 2000.

[6] I. Babuška and T. Strouboulis. *The Finite Element Method and Its Reliability.* Oxford University Press, 2001.

[7] I. Babuška and W. C. Rheinboldt. *A-posteriori* error estimates for the finite element method. *International Journal for Numerical Methods in Engineering*, 12(10):1597–1615, 1978.

[8] I. Babuška and W. C. Rheinboldt. Error estimates for adaptive finite element computations. *SIAM Journal on Numerical Analysis*, 15(4):736–754, 1978.

[9] T. C. Bailey, M. Colburn, B. J. Choi, A. Grot, J. G. Ekerdt, and C. G. Willson S. V. Sreenivasan. Step and Flash Imprint Lithography. In *Alternative Lithography: Unleashing the Potentials of Nanotechnology*, Nanostructure Science and Technology, pages 117–138. Springer, 2003. ISBN: 0306478587.

[10] T. C. Bailey, S. C. Johnson, S. V. Sreenivasan, J. G. Ekerdt, C. G. Willson, and D. J. Resnick. Step and Flash Imprint Lithography: An efficient nanoscale printing technology. *Journal of Photopolymer Science and Technology*, 15(3):481–486, 2002.

[11] S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith. PETSc users manual. Technical Report ANL-95/11 - Revision 2.1.1, Mathematics and Computer Science Division, Argonne National Laboratory, 2001.

[12] R. E. Bank and A. Weiser. Some *a posteriori* error estimators for elliptic partial differential equations. *Mathematics of Computation*, 44(170):283–301, 1985.

[13] R. H. Bartels and G. W. Stewart. Solution of the matrix equation $AX + XB = C$. *Communications of the ACM*, 15(9):820–826, 1972.

[14] P. T. Bauman. *Adaptive Multiscale Modeling of Polymeric Materials Using Goal-Oriented Error Estimation, Arlequin Coupling, and Goals Algorithms*. PhD thesis, The University of Texas at Austin, 2008.

[15] P. T. Bauman, J. T. Oden, and S. Prudhomme. Adaptive multiscale modeling of polymeric materials with Arlequin coupling and Goals algorithms. *Computer Methods in Applied Mechanics and Engineering*, 198:799–818, 2009.

[16] P. T. Bauman, S. Prudhomme, and J. T. Oden. On the application of the Arlequin Method to the coupling of particle and continuum models. Technical Report ICES-07-28, Institute for Computational Engineering and Sciences, The University of Texas at Austin, October 2007.

[17] R. Becker and R. Rannacher. Weighted *a posteriori* error control in FE methods. In *ENUMATH-95, Paris, 1995, Proc. ENUMATH-97, World Sci. Pub.*., 1995.

[18] A. Ben-Israel and A. Charnes. Contributions to the theory of generalized inverses. *Journal of the Society for Industrial and Applied Mathematics*, 11(3):667–699, 1963.

[19] A. Ben-Israel and T. N. E. Greville. *Generalized Inverses: Theory and Applications*. Springer, 2003.

[20] S. Benson, L. C. McInnes, and J. J. Moré. TAO users manual. Technical Report ANL/MCS-TM-242, Mathematics and Computer Science

Division, Argonne National Laboratory, 2000.

[21] M. Berezhnyy and L. Berlyand. Continuum limit for three-dimensional mass-spring networks and discrete Korn's inequality. *Journal of the Mechanics and Physics of Solids*, 54:635–669, 2006.

[22] A. Berman and R. J. Plemmons. Cones and iterative methods for best least squares solutions of linear systems. *SIAM Journal on Numerical Analysis*, 11(1):145–154, 1974.

[23] L. T. Biegler, T. F. Coleman, A. R. Conn, and F. N. Santosa, editors. *Large-Scale Optimization with Applications: Part III: Molecular Structure and Optimization*. Springer, 1997.

[24] Å. Björck. *Numerical Methods for Least Squares Problems*. SIAM, Philadelphia, 1996.

[25] Yu. E. Boyarintsev. *Methods of Solving Singular Systems of Ordinary Differential Equations*. Wiley, Chichester, 1992.

[26] F. Brezzi, L.P. Franca, T. J. R. Hughes, and A. Russo. $b = \int g$. *Computer Methods in Applied Mechanics and Engineering*, 145:329–229, 1996.

[27] R. L. Burns, S. C. Johnson, G. M. Schmid, E. K. Kim, M. D. Dickey, J. Meiring, S. D. Burns, N. A. Stacey, C. G. Willson, D. C., Yi Wei, P. Fejes, K. A. Gehoski, D. P. Mancini, K. J. Nordquist, W. J. Dauksher, and D. J. Resnick. Mesoscale modeling for SFIL simulating polymerization kinetics and densification. In R. S. Mackay, editor, *Emerging*

Lithographic Technologies VIII. Edited by Mackay, R. Scott. Proceedings of the SPIE, Volume 5374, pp. 348-360 (2004)., volume 5374 of *Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference*, pages 348–360, May 2004.

[28] D. Cioranescu and P. Donato. *An Introduction to Homogenization*. Oxford University Press, 2000.

[29] M. Colburn, S. C. Johnson, M. D. Stewart, S. Damle, T. C. Bailey, B. Choi, M. Wedlake, T. B. Michaelson, S. V. Sreenivasan, J. G. Ekerdt, and C. Grant Willson. Step and Flash Imprint Lithography: A new approach to high-resolution patterning. In Y. Vladimirsky, editor, *Proceedings of SPIE/Emerging Lithographic Technologies III*, volume 3676, pages 379–389, Santa Clara, 3 1999.

[30] M. Colburn, I. Suez, B. J. Choi, M. Meissl, T. Bailey, S. V. Sreenivasan, J. G. Ekerdt, and C. G. Willson. Characterization and modeling of volumetric and mechanical properties for Step and Flash Imprint Lithography photopolymers. *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, 19(6):2685–2689, 2001.

[31] E. Collister. Stress-strain experiments on PMMA. private communication, 2006.

[32] Z. Cui. *Micro-Nanofabrication: Technologies and Applications*. Springer, 2006. ISBN: 3540289224.

[33] T. A. Davis. CHOLMOD users' guide. Technical report, University of Florida, 2005.

[34] T. A. Davis. Multifrontal multithreaded rank-revealing sparse QR factorization. Technical report, University of Florida, 2009.

[35] L. Demkowicz. 2d hp-adaptive finite element package (2dhp90) version 2.0. Technical Report ICES-02-06, Institute for Computational Engineering and Sciences, The University of Texas at Austin, January 2002.

[36] L. Demkowicz. Projection-based interpolation. Technical Report ICES-04-03, Institute for Computational Engineering and Sciences, The University of Texas at Austin, February 2004.

[37] L. Demkowicz. *Computing with hp-ADAPTIVE FINITE ELEMENTS: Volume I: One and Two Dimensional Elliptic and Maxwell Problems*. Chapman & Hall/CRC Press, 2006.

[38] L. Demkowicz, W. Rachowicz, and Ph. Devloo. A fully automatic *hp*-adaptivity. *Journal of Scientific Computing*, 17:127–155, 2002.

[39] L. Demkowicz, W. Rachowicz, D. Pardo, M. Paszynski, J. Kurtz, and A. Zdunek. *Computing with hp-ADAPTIVE FINITE ELEMENTS: Volume II Frontiers: Three Dimensional Elliptic and Maxwell Problems with Applications*. Chapman & Hall/CRC Press, 2007.

[40] H. Ben Dhia and G. Rateau. The Arlequin method as a flexible engineering design tool. *International Journal for Numerical Methods in Engineering*, 62(11):1442–1462, 2005.

[41] M. D. Dickey, R. L. Burns, E. K. Kim, S. C. Johnson, N. A. Stacey, and C. G. Willson. Study of the kinetics of Step and Flash Imprint Lithography photopolymerization. *AIChE Journal*, 51(9):2547–2555, 2005.

[42] M. D. Dickey and C. G. Willson. Kinetic parameters for Step and Flash Imprint Lithography photopolymerization. *AIChE Journal*, 52(2):777–784, 2006.

[43] M. Dorobantu and B. Engquist. Wavelet-based numerical homogenization. *SIAM Journal on Numerical Analysis*, 35(2):540–559, 1998.

[44] W. E, X. Li, and E. Vanden-Eijnden. Some recent progress in multiscale modeling. In *Multiscale Modelling And Simulation*, Lectures Notes in Computational Science and Engineering, pages 3–22. Springer, 2003. ISBN: 3540211802.

[45] B. Engquist and O. Runborg. Wavelet-based numerical homogenization with applications. In T.J. Barth, T. F. Chan, and R. Haimes, editors, *Multiscale and Multiresolution Methods: Theory and Applications*, Lecture Notes in Computational Science and Engineering. Springer, 2002.

[46] J. L. Ericksen. The Cauchy and Born hypothesis for crystals. In *Phase*

transformations and material instabilities in solids, pages 61–77. Academic Press, 1984.

[47] C. A. Felippa, K. C. Park, and M. R. Justino Filho. The construction of free-free flexibility matrices as generalized stiffness inverses. *Computers and Structures*, 68:411–418, 1998.

[48] G. H. Golub and C. F. van Loan. *Matrix Computations*. Johns Hopkins, third edition, 1996.

[49] U. Hornung. *Homogenization and Porous Media*. Springer, 1997.

[50] T. Y. Hou and X.-H. Wu. A multiscale finite element method for elliptic problems in composite materials and porous media. *Journal of Computational Physics*, 134:169–189, 1997.

[51] T. J. R. Hughes. Multiscale phenomena - Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods. *Computer Methods in Applied Mechanics and Engineering*, 127:387–401, 1995.

[52] W. Humphrey, A. Dalke, and K. Schulten. VMD – Visual Molecular Dynamics. *Journal of Molecular Graphics*, 14:33–38, 1996.

[53] C. Jhurani and L. Demkowicz. Dimensional reduction for a lattice-like mass-spring polymer model using *hp*-adaptivity. *Computer Methods in Material Science*, 6:161–170, 2006.

[54] C. Jhurani and L. Demkowicz. Derivation of continuum strain energy density of a polymer lattice with central pair potentials. Conference presentation, Ninth U.S. National Congress on Computational Mechanics, San Francisco, CA, USA, 2007.

[55] C. Jhurani, L. Demkowicz, J. T. Oden, S. Prudhomme, and P. T. Bauman. Dimensional reduction for molecular statics problems using *hp*-adaptivity. Conference presentation, Seventh World Congress on Computational Mechanics, Los Angeles, CA, USA, 2006.

[56] V. V. Jikov, S. M. Kozlov, and O. A. Oleĭnik. *Homogenization of differential operators and integral functionals*. Springer-Verlag, Berlin, 1994.

[57] S. Johnson, R. Burns, E. K. Kim, M. Dickey, G. Schmid, J. Meiring, S. Burns, C. G. Willson, D. Convey, Y. Wei, P. Fejes, K. Gehoski, D. Mancini, K. Nordquist, W. J. Dauksher, and D. J. Resnick. Effects of etch barrier densification on Step and Flash Imprint Lithography. *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, 23(6):2553–2556, 2005.

[58] S. Knapek. Matrix-dependent multigrid homogenization for diffusion problems. *SIAM Journal on Scientific Computing*, 20(2):515–533, 1999.

[59] J. Kurtz and L. Demkowicz. A fully automatic *hp*-adaptivity for elliptic PDEs in three dimensions. *Computer Methods in Applied Mechanics and Engineering*, 196:3534–3545, 2007.

[60] C. R. K. Marrian and D. M. Tennant. Nanofabrication. *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, 21(5):S207–S215, 2003.

[61] J. E. Meiring. *Mesoscale Simulation of the Photoresist Process and Hydrogel Biosensor Array Platform Indexed by Shape.* PhD thesis, The University of Texas at Austin, 2005.

[62] R. E. Miller and E. B. Tadmor. The Quasicontinuum method: Overview, applications, and current directions. *Journal of Computer-Aided Design*, 9:203–239, 2002.

[63] E. H. Moore. On the reciprocal of the general algebraic matrix. *Bull. Amer. Math. Soc*, 26:394–395, 1920.

[64] J. J. More and Z. Wu. Issues in large-scale global molecular optimization. In L. T. Biegler et al., editor, *Large-scale Optimization with Applications*, volume 94, pages 99–122, 1997.

[65] J. David Moulton, Jr. Joel E. Dendy, and James M. Hyman. The black box multigrid numerical homogenization algorithm. *Journal of Computational Physics*, 142(1):80–108, 1998.

[66] M. Zuhair Nashed. Perturbations and approximations for generalized inverses and linear operator equations. In M. Zuhair Nashed, editor, *Generalized Inverses and Applications*, pages 325–396, 8 1976.

[67] M. Zuhair Nashed and L. B. Rall. Annotated bibliography on generalized inverses and applications. In M. Zuhair Nashed, editor, *Generalized Inverses and Applications*, pages 771–1041, 8 1976.

[68] J. T. Oden, L. F. Demkowicz, T. Stroubolis, and P. Devloo. Adaptive methods for problems in solid and fluid mechanics. In I. Babuška et al., editor, *Accuracy Estimates and Adaptive Refinements in Finite Element Computations*, 1986.

[69] J. T. Oden and K. S. Vemaganti. Estimation of local modeling error and goal-oriented adaptive modeling of heterogeneous materials. *Journal of Computational Physics*, 164:22–47, October 2000.

[70] V. Sima P. Benner. Solving linear matrix equations with SLICOT. In *Proceedings of the European Control Conference*, Cambridge, UK, 9 2003.

[71] D. Pardo, L. Demkowicz, C. Torres-Verdin, and L. Tabarovsky. A goal oriented *hp*-adaptive finite element strategy with electromagnetic applications. Part I: Electrostatics. *International Journal for Numerical Methods in Engineering*, 65(8):1269–1309, 2006.

[72] M. Paszynski, A. Romkes, E. Collister, J. Meiring, L. F. Demkowicz, and C. G. Willson. On the modeling of Step-and-Flash Imprint Lithography using molecular statics models. Technical Report ICES-05-38, Institute for Computational Engineering and Sciences, The University of Texas at Austin, September 2005.

[73] R. Penrose. A generalized inverse for matrices. *Proc. Cambridge Philos. Soc.*, 51:406–413, 1955.

[74] S. Prudhomme and J. T. Oden. On goal-oriented error estimation for elliptic problems: application to the control of pointwise errors. *Computer Methods in Applied Mechanics and Engineering*, 176:313–331, 1999.

[75] D. J. Resnick, G. Schmid, M. Miller, G. Doyle, C. Jones, and D. LaBrake. Step and Flash Imprint Lithography template fabrication for emerging market applications. In H. Watanabe, editor, *Proceedings of SPIE Photomask and Next-Generation Lithography Mask Technology XIV*, volume 6607, page 66070T, 5 2007.

[76] C.-S. Shao, R. Byrd, E. Eskow, and R. B. Schnabel. Global optimization for molecular clusters using a new smoothing approach. *Journal of Global Optimization*, 16:167–196, 2000.

[77] J. Sherman and W. J. Morrison. Adjustment of an inverse matrix corresponding to changes in the elements of a given column or a given row of the original matrix. *Annals of Mathematical Statistics*, 20:621, 1949.

[78] S. Spagnolo. Sul limite delle soluzioni di problemi di Cauchy relativi all'equazione del calore. *Annali della Scuola Normale Superiore di Pisa*, 21:657–699, 1967.

[79] A. N. Tikhonov. Regularization of incorrectly posed problems. *Soviet Math.*, 4:1624–1627, 1963.

[80] J. Zhang W. Xing, Q. Zhang and Q. Wang. Some geometric properties of Lyapunov equation and LTI system. *Automatica*, 37:313–316, 2001.

[81] S. P. Xiao and T. Belytschko. A bridging domain method for coupling continua with molecular dynamics. *Computer Methods in Applied Mechanics and Engineering*, 193:1645–1669, 2004.

[82] T. I. Zohdi and P. Wriggers. *Introduction to Computational Micromechanics*. Springer, 2005.

Vita

Chetan Kumar Jhurani was born in Ujjain, India on November 13, 1978, the son of Jaishree Jhurani and Nanak Jhurani. He studied at the Choithram School in Indore from 1982 to 1996. He received the degree of Bachelor of Technology in Mechanical Engineering from Indian Institute of Technology, Bombay in 2000. In his third year, he worked as an intern at the Center for Atmospheric and Oceanic Sciences in Indian Institute of Science, Bangalore. After graduating, he was a software engineer with Geometric Software, a computer aided design company in Pune, for three years and McAfee Inc., a network security company in Bangalore, for one year.

While working in Pune he met Shilpa Gulati, a student in the Department of Mechanical Engineering in Indian Institute of Technology, Bombay. They married on August 5, 2004. The same month, he joined the University of Texas at Austin to pursue his doctoral studies in the Computational and Applied Mathematics (CAM) program in the Institute for Computational Engineering and Sciences (ICES). He was awarded the Preemptive Fellowship to incoming graduate students by University of Texas at Austin. Chetan has accepted a job with CGGVeritas in Houston where he will be working as a Seismic Imager.

Permanent address: c/o Mr. Suresh Tirthani
209 Bairathi Colony No 2
Indore MP 452001
India

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.